PCT

/ORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ^{6} :
G03F 9/00, G06F 17/00, 17/50

A1

(11) International Publication Number: WO 99/14637

(43) International Publication Date: 25 March 1999 (25.03.99)

(21) International Application Number: PCT/US98/19439

(22) International Filing Date: 17 September 1998 (17.09.98)

(30) Priority Data:

08/931,921	17 September 1997 (17.09.97)	US
60/059,306	17 September 1997 (17.09.97)	US
60/069,549	12 December 1997 (12.12.97)	US
09/130,996	7 August 1998 (07.08.98)	US
Not fumished	16 September 1998 (16.09.98)	US
Not fumished	16 September 1998 (16.09.98)	US
Not fumished	16 September 1998 (16.09.98)	US

(63) Related by Continuation (CON) or Continuation-in-Part (CIP) to Earlier Applications

US	60/059,306 (CIP)
Filed on	17 September 1997 (17.09.97)
US	08/931,921 (CIP)
Filed on	17 September 1997 (17.09.97)
US	60/069,549 (CIP)
Filed on	12 December 1997 (12.12.97)
US	09/130,996 (CIP)
Filed on	7 August 1998 (07.08.98)
US	Not fumished (CIP)
Filed on	16 September 1998 (16.09.98)
US	Not fumished (CIP)
Filed on	16 September 1998 (16.09.98)
US	Not furnished (CIP)
Filed on	16 September 1998 (16.09.98)

(71) Applicant *(for all designated States except US)*: NUMERICAL TECHNOLOGIES, INC. [US/US]; Suite 100, 2630 Walsh Avenue, Santa Clara, CA 95051 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: CHANG, Fang-Cheng [US/US]; 2434 Rock Street #9, Mountain View, CA 94043 (US). WANG, Yao-Ting [-/US]; 970 Corte Madera Avenue #311, Sunnyvale, CA 94086 (US). PATI, Yagyensh, C. [US/US]; 816 Amber Lane, Los Altos, CA 94024 (US).

(74) Agent: RICHARDSON, Kent, R.; Wilson Sonsini Goodrich & Rosati, 650 Page Mill Road, Palo ALto, CA 94304-1050 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: DATA HIERARCHY LAYOUT CORRECTION AND VERIFICATION METHOD AND APPARATUS

(57) Abstract

A method and apparatus for the correction of integrated circuit layouts for optical proximity effects which maintains the original true hierarchy of the original layout is provided. Also provided is a method and apparatus for the design rule checking of layouts which have been corrected for optical proximity effects. The OPC correction method comprises providing a hierarchically described integrated circuit layout as a first input (205), and a particular set of OPC correction criteria as a second input (260). The integrated circuit layout is then analyzed to identify features of the layout which meet the provided OPC correction criteria (210, 240). After the areas on the mask which need correction have been identified (310), optical proximity correction data (320) is generated in response to the particular set of correction criteria. Finally, a first program data is generated which stores the generated optical proximity correction data in a hierarchical structure (320) that corresponds to the hierarchical structure of the integrated circuit layout (310). As the output correction data is maintained in true hierarchical format, layouts which are OPC corrected according to this method are able to be processed through conventional design rule checkers with no altering of the data.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

DATA HIERARCHY LAYOUT CORRECTION AND VERIFICATION METHOD AND APPARATUS

1. Related Applications

This application is a continuation of, and claims benefit of the filing date of, United States patent application entitled "Method and Apparatus for Data Hierarchy Maintenance in a System for Mask Description," filed on September 16, 1998, invented by Fang-Cheng Chang, Yao-Ting Wang and Yagyensh C. Pati. Further, this application relates to, claims benefit of the filing date of, and incorporates by reference, the United States provisional patent application entitled "Data Hierarchy Advanced Mask Correction and Verification Method and Apparatus" having serial number 60/069,549, filed on December 12, 1997, invented by Fang-Cheng Chang, Yao-Ting Wang and Yagyensh C. Pati, and for which the present application is the utility application thereof. Each of the aforementioned patents are assigned to the assignee of the present invention.

Further, this application relates to and incorporates by reference, the United States provisional patent application entitled, "Mask Verification, Correction, and Design Rule Checking" having serial number 60/059,306, filed September 17, 1997, and invented by Fang-Cheng Chang, Yao-Ting Wang and Yagyensh C. Pati and United States patent application entitled, "Mask Verification, Correction, and Design Rule Checking", filed September 16, 1998, and invented by Fang-Cheng Chang, Yao-Ting Wang and Yagyensh C. Pati. This application also relates to and incorporates by reference the United States patent application entitled "Visual Inspection and Verification System", filed August 7, 1998, and invented by Fang-Cheng Chang, Yao-Ting Wang, Yagyensh C. Pati, and Linard Karklin. This application also relates to and incorporates by reference, the United States patent application entitled, "Phase Shifting Circuit Manufacture Method and Apparatus" having serial number

08/931,921, filed September 17, 1997, and invented by Yao-Ting Wang and Yagyensh C. Pati. Each of the aforementioned patents are assigned to the assignee of the present invention.

2. The Background of the Invention

a. The Field of the Invention

This invention relates to the field of integrated circuit manufacturing. In particular, the invention relates to the concepts and implementation techniques to make fast and efficient integrated circuit layout correction and verification possible.

b. Description of Related Art

In designing an integrated circuit (IC), engineers typically rely upon computer simulation tools to help create a circuit schematic design consisting of individual devices coupled together to perform a certain function. To actually fabricate this circuit in a semiconductor substrate the circuit must be translated into a physical representation, or layout, which itself can then be transferred onto the silicon surface. Again, computer aided design (CAD) tools assist layout designers in the task of translating the discrete circuit elements into shapes which will embody the devices themselves in the completed IC. These shapes make up the individual components of the circuit, such as gate electrodes, field oxidation regions, diffusion regions, metal interconnections, and so on.

The software programs employed by these CAD systems are usually structured to function under a set of predetermined design rules in order to produce a functional circuit. Often, these rules are determined by certain processing and design limitations. For example, design rules may define the space tolerance between devices or interconnect lines so as to ensure that the devices or lines do not interact with one another in any unwanted manner. Design rule limitations are frequently referred to as critical dimensions. A critical dimension of a circuit is commonly defined as the smallest width of a line or the smallest space between two lines. Consequently, the critical dimension determines the overall size and density of the IC. In present IC technology, the smallest critical dimension for state-of-the-art circuits is approximately 0.25 microns for line widths and spacings.

Once the layout of the circuit has been created, the next step to manufacturing the integrated circuit (IC) is to transfer the layout onto a semiconductor substrate. Optical lithography is a well known process for transferring geometric shapes onto the surface of a silicon wafer. The optical lithography process generally begins with the formation of a photoresist layer on the top surface of a semiconductor wafer. A mask having fully light non-transmissive opaque regions, which are usually formed of chrome, and fully light transmissive clear regions, which are usually formed of quartz, is then positioned over the photoresist coated wafer. Light is then shone on the mask via a visible light source or an ultra-violet light source. The light is focused to generate a reduced mask image on the wafer typically using an optical lens system which contains one or several lenses, filters, and or mirrors. This light passes through the clear regions of the mask to expose the underlying photoresist layer, and is blocked by the opaque regions of the mask, leaving that underlying portion of the photoresist layer unexposed. The exposed photoresist layer is then developed, typically through chemical removal of the exposed/non-exposed regions of the photoresist layer. The end result is a semiconductor wafer coated with a photoresist layer exhibiting a desired pattern which defines the geometries, features, lines and shapes of that layer. This pattern can then be used for etching underlying regions of the wafer.

Besides the aforementioned design rules, the resolution value of the exposure tool used in optical lithography also places limits on the designers of integrated circuit layouts. The resolution for an exposure tool is defined as the minimum feature that the exposure tool can repeatedly expose onto the wafer. Currently, the resolution for most advanced optical exposure tools is around 0.25 micron. As the critical dimensions of the layout become smaller and approach the resolution value of the lithography equipment, the consistency between the mask and the actual layout pattern developed in the photoresist is significantly reduced. Specifically, it is observed that differences in pattern development of circuit features depends upon the proximity of the features to one another.

With these limitations on IC design in mind, we note the data describing an IC pattern is usually represented in a condensed hierarchical fashion such as in a GDS-II data file. At the higher levels of pattern representation hierarchy, features are represented in a conceptual manner. For instance, a memory array may be described as having a given cell repeated for a certain number of rows and columns. The next lower level in the hierarchy might describe the basic memory cell, comprised of subcells A and B. Finally, at the lowest level, the most primitive subcells contain geometric primitives-rectangles and polygons. In order to generate a physical mask, the hierarchical data must first be flattened, enumerating every geometric instance described in the hierarchy. Flattening of the hierarchy typically results in several orders of magnitude increase in the size of data storage required to represent the pattern.

Since flattening the hierarchy results in such a large increase in the size of the file representing a given IC design, it is desirable to flatten the hierarchy at the latest point in the manufacture of a mask, which, in the best case, is at the time the mask design is loaded into the EB machine prior to physical manufacture. Currently however, this flattening process takes place at an earlier stage in the production of masks for some complicated IC's. This is because the original mask design for a complicated IC is typically manipulated after the original design is completed in order to perform one of a number of operations on the design. These operations are performed because of the precision needed in the masks for complicated IC's as the critical dimensions of these IC's approach the resolution limits of optical lithography. Currently, these operations require some sort of flattening of the original design data in order to be performed -- resulting in an earlier than desired flattening of the design data.

These operations include the performance of logical operations, the generation of optical proximity corrections, the generation of phase shifting masks, and the design rule checking of masks that have undergone these operations.

In particular, nearly all modern integrated circuits of even limited complexity require that the original mask design be corrected for optical proximity effects in order that the desired image be accurately reproduced on a wafer after photolithography. Proximity effects occur when very closely spaced pattern features are lithographically transferred to a resist layer on a wafer. The light waves passing through the closely spaced features interact and, as a result, distort the final transferred pattern features. Another problem that occurs when feature sizes and spacing approach the resolution limit of the lithographic tool is that corners (concave and convex) tend to overexpose or underexpose due to a concentration or scarcity of energy at each of the corners. Other types of problems, such as over- or under-exposure of small features when large and small features are transferred from the same mask pattern, also occur.

Numerous methods have been developed to overcome the proximity effect problem. These methods include: precompensating mask line widths, varying photoresist layer thicknesses, using multi-layer photoresist processes, using electron beam imaging in conjunction with optical imaging, and finally, adding additional features to the original mask pattern to compensate for proximity effects. This last method is known as "Optical Proximity Correction" (OPC).

The additional features that are added to the original mask when OPC is utilized are typically sub-resolution (i.e. have dimensions less than the resolution of the exposure tool) and thus do not transfer to the resist layer. Instead, they interact with the original pattern so as to improve the final transferred pattern and compensate for proximity effects.

Currently there are several known OPC software implemented products available that adjust mask definitions to include OPC features. However, thus far, the available products have a number of limitations in terms of correctness, speed, data volume, and verification of the resultant OPC corrected mask design. For, the current products do not maintain the true hierarchical data format of the original mask design when the OPC features are added to the mask design. These products must first expand the original mask design to some type of a flattened data format prior to compensating by adding correction features. This causes the size of the resultant corrected design data file to increase severalfold, and thus slow down the process of OPC. Further, and more importantly, because they do not maintain the original true hierarchical data format of the mask design, it is extremely difficult and time consuming to verify currently known OPC corrected masks using conventional verification tools which require the same hierarchical data format as the original mask.

Therefore, what is desired is a method and apparatus for OPC correcting integrated circuit mask designs that solves the aforementioned problems of currently known systems with respect to correctness, speed, data volume, and verification of results.

3. A Summary of the Invention

As discussed above, currently known systems for the correction of and performance of logical operations on integrated circuit design layouts are not capable of preserving the original hierarchy of the design. This leads to several problems including a large increase in data, a reduction in processing speed, and the loss of the ability to quickly check processed designs for correctness using conventional verification tools.

Accordingly, in one embodiment of the present invention a method for the correction of integrated circuit layouts for optical proximity effects which maintains the hierarchy of the original layout is provided. The correction method comprises providing a hierarchically described integrated circuit layout as a first input, and a particular set of correction criteria as a second input. The integrated circuit layout is then analyzed to identify features of the layout which meet the provided particular correction criteria. After the areas on the mask which need correction have been identified, optical proximity correction data is generated in response to the particular set of correction criteria. Finally, a first program data is generated which stores the generated optical proximity correction data in a hierarchical structure that corresponds to the hierarchical structure of the integrated circuit layout.

In various alternative embodiments of the method, the generated first program data is stored on a computer readable media such as a hard disk drive or a server, and the integrated circuit layout is described by a GDS-II data file or other hierarchical description of the data. In another instance of the above embodiment, the first program data may also be described by a GDS-II data file or other hierarchical description of the data.

In further characterizations of the above method, the generated optical proximity correction data comprises data corresponding to the addition of serif features to the layout, and data corresponding to the correction of transistor gates. In a further embodiment of the above method, the particular set of correction criteria comprises a means for specifically identifying transistor gates in the layout. This instance is further characterized wherein the generated optical proximity correction data may comprise data corresponding to the addition of hammerhead features to the line ends of transistors or wires.

In an alternate embodiment, the method is further characterized by the additional step of combining the first program data with the data describing the integrated circuit mask to produce a second program data that describes a first corrected mask. This embodiment can be further characterized by adding another step which includes flattening the second program data to produce a third program data, and utilizing the third program data to produce an optically corrected lithographic mask.

This embodiment can be further characterized by the addition of a step of providing the second program data to a design rule checker to determine whether the first corrected layout falls within a set of design rules associated with the integrated circuit.

In another instance, this embodiment can be further characterized by the additional steps of providing a set of providing a set of layout accuracy parameters and comparing the first corrected layout to the design accuracy parameters. These added steps further include providing a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules and applying the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within the set of layout accuracy parameters.

This embodiment can be further characterized by adding the steps of providing the second corrected layout to a design rule checker, operating the design rule checker, and determining whether the second corrected layout falls within a set of design rules associated with the integrated circuit. Alternatively, the above embodiment can be further characterized wherein the step of comparing the first corrected layout to the design accuracy parameters further comprises providing the second program data to a checker device which produces a simulated image of the exposure that the first corrected layout would produce, providing the actual image of the exposure that was designed for the integrated circuit, and measuring the difference between the simulated image and the actual image.

In another instance, the method may also be characterized in that the step of generating the first program may further include generating a plurality of delta planes corresponding to the plurality of cells. In this embodiment, each delta plane comprises data representative of the difference between a correction plane of the cell corresponding to the delta plane and the delta planes corresponding to the children cells of the cell corresponding to the delta plane. In this instance, the correction plane for each cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data.

An alternative embodiment of the basic method claim is also provided. This alternative provides for the steps of providing the integrated circuit layout as a first input and providing a particular set of correction criteria as a second input. Other steps include compiling the hierarchical tree structure, wherein compiling comprises generating a first correction layer for each cell of the plurality of cells in the hierarchical tree structure in response to the particular set of correction criteria. Also included is linking the hierarchical tree structure, wherein linking comprises modifying the correction layer of each cell to generate a delta plane for each cell such that the delta plane of each cell accounts for interaction between each of the cell's child cells and interaction between the cell's primitive geometry and each of the cell's child cells. Lastly, a first program data comprising the delta planes is provided, wherein the first program data is configured hierarchically such that it corresponds to the hierarchical tree structure of the integrated circuit layout.

In an alternate embodiment of this method an additional limitation is added to the above method wherein for each cell in the hierarchical tree structure the sum of the delta plane of the cell and the delta planes of the cell's child cells comprises a correction plane of the cell wherein the correction plane for the cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data. Other variations of this alternate method are similar to those discussed above with respect to the main method.

Lastly, the method steps of the above embodiments may in one instance be performed by a computer running a program which implements these steps wherein the program is stored on any appropriate computer storage media such as a hard disk drive or server.

The present invention, as summarized above with respect to method steps, may be alternatively characterized as an apparatus for the correction of integrated circuit layouts for optical proximity effects which maintains the hierarchy of the original layout. The apparatus includes, in one embodiment, a first input that receives the integrated circuit layout, a second input that receives a particular set of correction criteria, and a resource that analyzes the integrated circuit layout and identifies features that meet the particular set of correction criteria. Also provided is a resource that generates optical proximity correction data in response to the particular set of correction criteria for the features that meet the particular set of correction criteria and a resource that provides a first program data wherein the first program data comprises the optical proximity correction data configured in a hierarchical structure that corresponds to the hierarchical structure of the integrated circuit layout.

The features discussed above with respect to the method embodiment apply as well with respect to the apparatus embodiment. For instance, the apparatus embodiment can be altered to provide for the addition of serifs and transistor gate corrections to the correction data, and provide for a particular set of correction criteria which comprises a means for specifically identifying transistor gates.

In an alternate embodiment, the apparatus comprises an additional resource that combines the first program data with data describing the integrated circuit mask to provide a second program data that describes a first corrected layout. This embodiment can be further characterized by the addition of a resource that flattens the second program data to produce a third program data, and a resource that utilizes the third program data to produce an optically corrected lithographic mask. This embodiment can be further altered by the provision of a design rule checker which determines whether the first corrected layout falls within a set of design rules associated with the integrated circuit.

In another embodiment, still further elements are added, such as for instance, a third input that receives a set of layout accuracy parameters and a resource that compares the first corrected layout to the design accuracy parameters. Also provided is a resource that provides a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules and a resource that applies the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within the set of layout accuracy parameters. This embodiment can be further characterized by the addition of a design rule checker that receives the second corrected mask and determines whether the second corrected layout falls within a set of design rules associated with the integrated circuit.

This embodiment can be still further characterized by the addition of a resource that provides the second program data to a checker device which produces a simulated image of the exposure that the first corrected layout would produce. Also provided in this instance is a resource that provides the actual image of the exposure that was designed for the integrated circuit and a resource that measures the difference between the simulated image and the actual image.

Lastly, the apparatus of the above embodiments may in one instance be represented by a computer program product which comprises, in one instance, a computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout, wherein the data describing the integrated circuit layout comprises a hierarchical structure.

Although many details have been included in the description and the figures, the invention is defined by the scope of the claims. Only limitations found in those claims apply to the invention.

4. A Brief Description of the Drawings

The figures illustrate the invention by way of example, and not limitation. Like references indicate similar elements.

Fig. 1 illustrates a simple integrated circuit design layout and a hierarchical tree representation of the layout.

Fig. 2 illustrates a system level depiction of one embodiment of the invention.

Fig. 3 illustrates a simple representation of a typical hierarchical data file that would be output from the system of Fig. 2.

Fig. 4 illustrates in flowchart form, a method of performing a logical or arithmetic operation on a hierarchical integrated circuit design in which the hierarchical structure of the design layout is maintained according to one embodiment of the invention.

Fig. 5 illustrates how the method of Fig. 4 would provide for a logical NOT operation on one of the parent cells of Fig. 1 according to one embodiment of the invention.

Fig. 6 illustrates how the method of Fig. 4 would generate the delta plane of one of the parent cells of Fig. 1 for a logical NOT operation according to one embodiment of the invention.

Fig. 7 illustrates examples of optical proximity corrections that can be made to design layouts.

Fig. 8 is a depiction of a system for providing OPC correction to a design layout in accordance with one embodiment of the invention.

Fig. 9 illustrates how one embodiment of the system of Fig. 8 would provide for OPC correction of the primitive geometries of one of the cells of

Fig. 1.

Figs. 10(a)-(b) illustrate how the method of Fig. 4 would generate a correction layer for overlap areas within one of the parent cells of Fig. 1 for an OPC operation according to one embodiment of the invention.

Fig. 11 illustrates how the method of Fig. 4 would generate the delta plane of one of the parent cells of Fig. 1 for an OPC operation according to one embodiment of the invention.

Fig. 12 illustrates a further method for providing OPC correction to an integrated circuit design layout using one embodiment of the present invention.

Fig. 13 illustrates an example screen snapshot of an input design layout from a computer system executing one embodiment of the invention to provide OPC correction of the input design layout.

Fig. 14 illustrates an example screen snapshot of the final output from a computer system executing one embodiment of the invention to provide OPC correction to the input design of Fig. 13.

Fig. 15 illustrates a zoomed in example screen snapshot of the final output of Fig. 14.

Fig. 16 illustrates an example screen snapshot of a -1 OPC correction layer from a computer system executing one embodiment of the invention to provide OPC correction.

Fig. 17 illustrates an example screen snapshot of a +1 OPC correction layer from a computer system executing one embodiment of the invention to provide OPC correction.

Fig. 18 illustrates an example screen snapshot of a -2 OPC correction layer from a computer system executing one embodiment of the invention to provide OPC correction.

Fig. 19 illustrates an example screen snapshot of an individual cell that has been OPC corrected by a computer system executing one embodiment of the invention.

Although many details have been included in the description and the figures, the invention is defined by the scope of the claims. Only limitations found in those claims apply to the invention.

5. The Detailed Description

As described above, in the manufacture of photolithography masks, it is advantageous to flatten the data representing the IC design at the latest point in the actual manufacture of the masks. However, this flattening takes place earlier than desired in some instances. This is because the original layout for a complicated IC is typically manipulated after the original design is completed in order to perform one of a number of operations on the design. These operations include the performance of logical operations, the generation of optical proximity corrections, the generation of phase shifting masks, and the design rule checking of masks that have undergone these operations. Currently, these operations require some sort of flattening of the original design data in order to be performed -- resulting in an earlier than desired flattening of the design data. This early flattening of data results in large increases in required data storage and a corresponding slow down in the performance of these operations. Further, because existing verification systems typically require the same input data hierarchies, operations which modify a design in a non-hierarchical manner make it difficult if not impossible to perform the important step of verifying these modified designs.

The present invention solves the above problems by providing for the performance of operations, such as OPC correction, on an input hierarchical IC design such that the original true hierarchy of the design is maintained. Various embodiments of the invention include computer systems for verifying and correcting masks for use in integrated circuit manufacturing, and for performance of logical operations on design layouts. These embodiments receive hierarchical mask definition data defining the look of a particular mask. These embodiments then generate an output set of data. In one embodiment, this output data includes OPC corrected mask definitions. Other embodiments of the invention include actual masks generated using systems that perform OPC correction or mask verification techniques. Still other embodiments of the invention include computer readable media (e.g., hard disks, CDs, and other computer program storage tools) having computer programs that implement the OPC correction or mask verification techniques.

Before turning to a description of the present invention with respect to the figures, an overview of one embodiment of the concept of the present invention is provided. Thus, one embodiment of the invention utilizes a hierarchy preserver to receive a hierarchical definition of a layout and to generate one or more additional data layers that hierarchically include correction information provided by an engine which performs the operation on the design layout. These additional layers are then stored such that they are associated with each node in the hierarchic definition of that layer.

The following definitions are used in this specification: A correction plane is associated with any node (cell) in the hierarchy such that, by applying the correction plane to the flattened node, the output is the corrected design for that node. A delta plane is essentially the difference between a node's correction plane and the sum of all its immediate children's delta planes. Thus, the correction plane of a cell is equal to the delta plane for that cell plus the delta planes of the immediate child cells of that cell. Since the leaf cells of the hierarchy have no child cells, the correction plane for any leaf cell is equal to the delta plane of that leaf cell. In this manner, in one instance of the invention, the correction plane for each cell need not be stored since the overall correction to the layout can be provided by only storing the delta plane for each cell in the hierarchy.

The basic idea behind one embodiment of the present invention is described in two stages which comprise compilation and linking. In the compilation stage, corrections are generated for all geometry primitives in the hierarchy in accordance with the operation to be performed on the design layout. At the link stage, extra corrections due to the optical overlap of a parent cell's child cells and the parent cell's primitive geometry will be made. Therefore, only the additional correction is stored.

A delta algorithm computes the delta/additional information by considering only the children cell overlaps and the overlap between the parent's geometry and child cells. Only these areas are considered because only the overlaps would contribute to the additional correction change needed for the parent cell. The overlap area is not limited to purely geometry overlap, but also includes proximity overlap. By employing a more general definition, all proximity effects / corrections can be taken into account. The output of the delta algorithm for a cell will be now called its delta plane. The leaves of the hierarchical tree thus have delta planes equal to their correction planes.

At compile time, the correction planes for all leaves can be generated by providing the flattened data describing the geometry primitives for each leaf to an operation engine which performs the desired operation on the provided flattened data. At link time, if there is no subcell overlap, then the correction plane for this parent cell equals to the sum of its children's delta planes (and as described above their would be no additional delta plane information to be stored for this parent cell). If there is an overlap, the overlap area is flattened, and an intermediate correction plane for the flattened overlap area is generated. Subsequently, this intermediate correction area is used to subtract the sum of all correction planes of its children, and the difference is the delta plane which is stored hierarchically to correspond to the cell being linked.

The current GDS-II, and most other design database formats describing a full layout, include placing different mask and chip levels on separate layers. What is being introduced in various embodiments of the invention is a twist on the layer concept--that an arithmetic layer capable of both logical (e.g., XOR, AND) and arithmetic operations can be based upon. For instance, with respect to an OPC operation, a correction layer representing a particular OPC feature can be based on an arithmetic layer such that for example, "-1" means a negative serif, a "+1" means a positive serif, and "-2" means end-butting where the overlap is infinitesimal in one direction. During linking, all correction layers are arithmetically generated using an algorithm to compute the incremental or differential corrections throughout the structure. These delta planes, or arithmetic layers, are exposed in the database format as distinct layers (e.g., +1, -1, -2, etc. mapping to layers 1, 2, and 3). This allows the final correction layer for a parent cell to equal to the parent cell's delta plane and the incremental sum of all the delta planes of the parent cell's children and grandchildren and great-grandchildren, and so on, from the leaves' compile-time correction layer.

Hierarchical data management is also possible in the generation of corrections in an alternate embodiment of the invention in which the delta algorithm or arithmetic layers discussed above are not used. In this alternate embodiment, instead of taking and storing the difference between the correction layers of a parent cell and its children cells, a logical operation can be used to compare the correction between the parent and its children, and the "logical" instead of "arithmetic" difference is then stored at the parent cell.

Thus, as summarized above, the present invention provides a method and apparatus for data hierarchy maintenance in a system for mask description. A detailed description of preferred embodiments of the present invention is now provided with respect to the figures in which Fig. 1 illustrates a simple integrated circuit design layout 100 and a hierarchical tree representation of the layout 110. The circuit layout 100 comprises a final cell A which comprises Parent Cells B, C, and D. Parent Cell C comprises identical cells G1, G2, G3, G4, G5, and G6. Parent Cell D comprises cell H and identical cells I1 and I2. Parent cell B comprises identical Parent Cells E1 and E2, and identical Parent cells F1 and F2. Parent Cell E1 comprises leaf cells J1 and K1 which comprise the primitive geometric structures illustrated in Fig. 1. Parent Cell E2 comprises Leaf cells J2 and K2 which comprise the same primitive geometric structures as cells J1 and K1. Parent Cell F1 comprises Leaf cells L1 and M1 which comprise the primitive geometric structures illustrated in Fig. 1. Parent Cell F2 comprises Leaf cells L2 and M2 which comprise the same primitive geometric structures as cells J1 and K1. The hierarchical tree layout 110 illustrates the aforementioned cells in a tree format with the leaf cells at the bottom of the tree and with the final cell A at the top of the tree. Each of the leaf cells is also sometimes referred to herein as a leaf node or a child cell, while each of the cells above the leaf nodes is sometimes referred to herein as a parent cell or simply a node. The integrated circuit design layout 100 of Fig. 1 is provided as a reference IC design with respect to which the embodiments of the present invention are described below. The simple IC illustrated in Fig. 1 is used for example only, as the embodiments of the invention described below are capable of being applied to any IC which is described in a hierarchical format.

Fig. 2 illustrates in block diagram form a system incorporating one embodiment of the present invention. The system described is one in which a logical or arithmetic operation can be performed on a hierarchically described input IC design such that the resultant modified IC design retains the original true hierarchy of the input design. The basic elements of one embodiment of the system comprise a hierarchy preserver 210 and an operation engine 240. The hierarchy preserver 210 comprises a compiler 220 and a linker 230.

The hierarchy preserver 210 of the system accepts hierarchical design data 205 which describes an integrated circuit design 200 as an input. The hierarchy preserver 210, in one embodiment, accepts hierarchical design data 205 in a GDS-II format. In other embodiments, the hierarchy preserver 210 accepts hierarchical design data 205 described in any hierarchical file format. The compiler 220 of the hierarchy preserver 210 acts in conjunction with the operation engine 240 to provide a correction data layer for the geometry primitives at each node of the design data 205. The generated correction data layers are representative of the changes to be made to the geometry primitives at each node in accordance with the operation being performed by the operation engine 240 as will be described more fully below. In one embodiment of the invention, the operation engine 240 performs a logical operation such as AND or NOT on the input design data 205. In another embodiment of the invention, the operation engine 240 performs optical proximity corrections on the input design data 205. In still another embodiment of the invention, the operation engine 240 performs design rule checking of the input design data 205.

After the compiler 220 has generated a correction data layer for each node of the input design data 205, the linker 230 acts in conjunction with the operation engine 240 to generate a delta plane for each node of the design. The delta plane for each cell is generated such that it is equal to the difference between the correction data layer information for the particular cell and the sum of all the correction data layers of the particular cell's children cells. In one embodiment, the delta plane for each cell is generated by a delta algorithm processed by the linker 230 which computes the delta/additional information by only considering overlaps within each cell. In one embodiment, these overlaps consist of the overlaps between a cell's children cells and any overlaps between a parent cell's own primitive geometry and that of it's children cells. In one embodiment, these overlap areas are not limited to purely geometry overlap, but also include proximity overlap. The process by which the linker 230 generates a delta plane for each node of the input design 205 will be described more fully below.

After the linker 230 has generated the delta planes, the hierarchy preserver 210 generates output data 250 which represents the input design 205 modified in accordance with the operation performed by the operation engine 240, where the output data 250 retains the original true hierarchy of the input design data 205. This output data 250 comprises the original unaltered hierarchical design data 205, and a hierarchical correction data file 260. The hierarchical correction data file 260 comprises the delta plane data for each node of the design data 205 such that when the design data 205 and the correction data 260 are combined a modified design is produced which represents the operation performed on the original design data 205 by the operation engine 240.

The hierarchical output data 250 can then be used for a number of purposes. First, it could be provided to the hierarchy preserver 210 on line 262 in order for a new logical or arithmetic operation to be performed on the output data 250. Further, since it is in hierarchical form, it can be provided to a conventional design rule checker 270 which accepts hierarchical data, in order that the new modified output design can be checked to verify that it meets the design rules for the particular integrated circuit being designed. Still further, the output data 250 can bc used in mask production 265 by combining the design data 205 with the correction data 260 to construct a final data layout 275, flattening this combined data layout 280, and providing this flattened data to an electron beam machine to generate the actual physical mask which embodies the modified design data 285.

The generation of correction data layers and delta planes for each of the nodes of the design data 205 will now be developed further. With reference to Fig. 1, one embodiment of the compiler 220 accesses the design data using a depth wise traverse in which each branch of the final parent cell is accessed in order, and in which each branch is accessed from its leaf nodes upwards. Thus referring to Fig. 1, this embodiment of the compiler 220 would access the nodes of the integrated circuit layout 100 in the following order: J1, K1, E1, L1, M1, F1, L2, M2, F2, J2, K2, E2, B, G1, G2, G3, G4, G5, G6, C, H, I1, I2, D, and A. As the compiler 220 traverses the tree it provides the flattened data corresponding to the primitive geometry of each cell to the operation engine 240. The operation engine 240 performs an operation on the flattened data and returns the results of this operation to the hierarchy preserver 210. For instance, with respect to Fig. 1, if leaf cell J1 were compiled, the operation engine 240 would return flattened data $J' = J + \Delta J$. In one embodiment, the amount of data storage is decreased by the hierarchy preserver 210 which solves the aforementioned equation for ΔJ, and stores the value of ΔJ as the correction layer for cell J. This process is repeated for every cell in the design until the entire tree is traversed. The design data 205 is then linked by the linker 230 in the following manner. The tree is again traversed in the manner described above, and for each cell the overlap area is found and flattened. The flattened overlap area is then input to the operation engine 240 which in turn operates on the data and returns it to the hierarchy preserver 210. The linker 230 utilizes the return data from the operation engine 240 to produce an intermediate correction layer which is used by the linker 230 to generate a delta plane for each cell. The generation of the delta plane will be more fully described below with respect to Figs. 6 and 10. The delta plane for each cell of the design is then stored in a hierarchical format corresponding to that of the input design data 205 in hierarchical correction data file 260.

In one embodiment of the invention as described in Fig. 2, the hierarchy preserver 210 may comprise a computer system executing program code stored on computer readable media that performs the functions of the compiler 220 and the linker 230. In one embodiment of the invention the operation engine 240 may also comprise a computer system executing program code stored on computer readable media. In one embodiment of the invention the hierarchy preserver 210 and the operation engine 240 comprise a single computer system executing a program code stored on computer readable media which performs the functions of the compiler 220 , linker 230, and operation engine 240 together. In another embodiment, the hierarchy preserver 210 and the operation engine 240 comprise either a single computer system executing two or more different program codes or multiple separate computer systems executing two or more different program codes, one code for the functions of the hierarchy preserver 210, and a separate code for the functions of the operation engine 240. In this embodiment, the hierarchy preserver 210 may selectively provide data to the operation engine 240 through an API. With this embodiment, the hierarchy preserver 210 of the present invention can be modified to communicate and operate with currently existing operation engines 240 to provide the advantages of hierarchical data output.

The computer readable media referred to above may comprise any computer storage tools including but not limited to hard disks, CDs, floppy disks, and server memory. The computer systems executing the program code may, in the case of both the operation engine 240 and the hierarchy preserver

210, comprise any suitable computer system including a desktop computer running a Windows NT operating system or a Sun Solaris workstation for instance.

Turning to Fig. 3, a simplified representation of a typical hierarchical data file that would be output from one embodiment of the system of Fig. 2 is illustrated. Hierarchical data file of correction data 320 represents a simplified version of the correction data that would be generated if the system of Fig. 2 were applied to operate on the simplified integrated circuit layout 100 of Fig. 1. As described above, hierarchical design data 205 is provided to hierarchy preserver 210 which operates in conjunction with operation engine 240 to provide hierarchical correction data 260. A simplified hierarchical data file of the design layout 310 is shown to illustrate the minimal effect the present invention has on the increase in data upon the performance of an operation. For, as shown, the hierarchical data file of correction data 320 is able to be stored in a structure which corresponds one to one with the input data file 310. This facilitates quick combination of the two data files 310 and 320 in order to perform other functions on the overall modified design such as mask production and design rule checking.

Note also that when a cell is traversed by the hierarchy preserver 210, the hierarchy preserver 210 determines whether or not that cell is identical to a cell that has already been traversed. If this is the case, then the hierarchy preserver does not take the processing time to directly determine a delta plane for that cell. Instead, the hierarchy preserver maintains the true hierarchy by providing a pointer to that cell's first instance of being defined. For instance, this is illustrated by the hierarchical data file of correction data 320 by cells F1 and F2 which are identical cell's as shown in Fig. 1. As described earlier, in one embodiment of the invention, the hierarchy preserver 210 traverses the design data 205 in a depth wise manner from the leaf nodes to the final parent cell. In this manner, F1 would be traversed before F2, and thus correction data ΔF1 would be generated and stored for this cell as indicated in file 320 by label 325.

When cell F2 is traversed however, only a pointer to the correction data for F1 is stored and no direct correction data is processed for F2. This is indicated by label 330. In this manner, both processing time and data volume are decreased.

Fig. 4 illustrates in flowchart form, a method of performing a logical or arithmetic operation on a hierarchical integrated circuit design in which the hierarchical structure of the design layout is maintained according to one embodiment of the invention. At its simplest level the method comprises a compiling process followed by a linking process. A hierarchical design data layout is provided at block 400 and the design tree is accessed at block 410 in the manner previously described with respect to Figs. 2 and 3. The compile process begins at block 415 wherein the hierarchical data for the first cell in the tree is obtained. Next, at block 425 it is determined whether or not the cell has been previously defined. If it has been previously defined, the obtained cell is associated with the previously defined correction data at block 427, and the next cell in the tree is obtained at block 415. If the cell has not been previously defined, the flattened primitive structure data of the cell is obtained at block 430 and provided to block 435 where an arithmetic or logical operation is performed on the flattened primitive data. The modified flattened primitive data is then provided to block 440, and this data is then processed at block 445 to separate the desired correction data as described earlier with respect to ΔJ of Fig. 2. The separated correction data is then stored in a hierarchical fashion corresponding to the original design data at block 450. At block 455, it is determined whether all of the cells have been traversed. If they have, the linking process begins at block 460, if not the compiling process continues at block 415 until such time when all the cells have been traversed and compiled.

The linking process begins in the same manner as the compile process with the accessing of the design tree at block 460. The process continues at block 465 wherein the hierarchical data for the first cell in the tree is obtained. Next, at block 470 it is determined whether or not the cell has been previously defined. If it has been previously defined, the obtained cell is associated with the previously defined correction data at block 427, and the next cell in the tree is obtained at block 465. If the cell has not been previously defined, the overlaps of the cell are determined at block 475 as discussed earlier with respect to Fig. 2. These overlap areas are then flattened at block 480 and the flattened data is provided to block 435 where the arithmetic or logical operation is performed on the flattened data as discussed previously. This operated upon flattened data is then used to generate an intermediate correction layer at block 483, and at block 485 a delta plane for the cell is generated which is then stored in a hierarchical data format at block 490. The delta plane is the only data which needs to be kept for each cell in the tree. For, as discussed above, given a parent cell and its child cells, the difference between the correction information for the parent cell and the sum of all of its children's correction data is equal to the delta plane. Thus, it follows then that the leaves of the hierarchical tree have delta planes equal to their correction planes determined at compilation. At block 495 it is determined whether or not all the cells in the tree have been traversed. If so, then the process is stopped, and the output data may then be used for a variety of functions as discussed previously -- if not then the linking process continues at block 465 until all of the tree's cells have been traversed.

Fig. 5 illustrates how a specific embodiment of the present invention would perform a logical operation upon Parent cell F1 of Fig.1. In this instance, suppose what was desired was to perform a logical NOT operation on Parent cell F1. The desired output is illustrated in Fig. 5 as F1(NOT). To perform this operation directly would entail performing a NOT operation on the flattened data representing Leaf L1 and a NOT operation on the flattened data representing M1 using the operation engine 240 of Fig. 2 in the manner described above with respect to block 435 of Fig. 4. The results of these operations could then be stored in a hierarchical manner such that the correction data is associated with the appropriate nodes. These results are illustrated in Fig. 5 as L1(NOT) and M1(NOT). Without the teachings of the present invention however, one can not obtain the desired F1(NOT) results simply by summing L1(NOT) and M1(NOT). This is illustrated by incorrect result 510 which is obtained by summing L1(NOT) and M1(NOT).

One embodiment of the present invention would operate to obtain the correct result F1(NOT) as follows. With reference to Fig. 2, the hierarchical design data 205, which in this simple example consists of the data representing Parent cell F1, is provided to the compiler 220 of hierarchy preserver 210. The compiler 220 provides the flattened data representing leaf L1 to the operation engine 240 which in this case performs a logical NOT operation on the supplied data and returns flattened data representative of the NOT of leaf L1. As described above, the compiler 220 then generates the correction data for L1 and stores this data in hierarchical correction data file 260. The same process is then repeated for leaf M1. As there is no primitive geometry associated with Parent cell F1, compilation of Parent F1 results in the generation of no correction data for F1. After the compilation of F1, the linker 230 operates to generate a delta plane 520 for Parent cell F1 during the linking stage as described more fully below with respect to Fig. 6. The delta plane 520 is generated such that, when summed with L1(NOT) and M1(NOT), the correct desired result F1(NOT) is obtained as shown in Fig. 5. This delta plane data is then stored hierarchically in hierarchical correction data file 260 such that it is associated with parent cell F1. This example is solely for illustration of the use of one embodiment of the present invention in the performance of a particular logical operation on a sample IC layout. As such, it is clear that this embodiment of the present invention could be used to perform any logical operation on any IC layout described in a hierarchical manner.

Fig. 6 illustrates heuristically how the delta plane 520 of Fig. 5 would be generated according to one embodiment of the present invention. With reference back to Fig. 2, after the operation engine 240 has performed a logical NOT operation on the flattened primitive data of leaf cells L1 and M1 respectively during the compile stage, the hierarchy preserver 210 operates during linking of Parent F1 to find the overlap areas within the parent cell and flatten these areas 600, which generates overlap area 640. This flattened data for overlap area 640 is then provided to operation engine 240 at block 610, and the NOT of the overlap 650 is generated by the operation engine 240. The logical NOT operation is then performed on the parent F1 to generate F1(NOT) in a flattened form at block 620. Lastly, the delta plane 520 is generated by taking the difference between the NOT of the overlap area 650 and flattened F1(NOT), and storing this delta data in hierarchical correction data file 260.

The use of the present invention in a system for generating OPC corrected layouts is now described. As stated previously, as the features of integrated circuit designs have become smaller and smaller, the resolution limits of optical lithography have had an increased effect on the exposure process. For instance, it has been observed that differences in pattern development of circuit features depend upon the proximity of the features to one another. Proximity effects occur when very closely spaced pattern features are lithographically transferred to a resist layer on a wafer. The light waves of the closely spaced features interact and, as a result, distort the final transferred pattern features. Another problem that occurs when feature sizes and spacing approach the resolution limit of the lithographic tool is that corners (concave and convex) tend to overexpose or underexpose due to a concentration or scarcity of energy at each of the corners. Other types of problems, such as over- or under-exposure of small features when large and small features are transferred from the same mask pattern, also occur.

Numerous methods have been developed to overcome the proximity effect problem. These methods include: precompensating mask line widths, varying photoresist layer thicknesses, using multi-layer photoresist processes, using electron beam imaging in conjunction with optical imaging, and finally, adding additional features to the original mask pattern to compensate for proximity effects. This last method is known as Optical Proximity Correction (OPC).

Fig. 7 illustrates examples of optical proximity corrections that can be made to design layouts. The additional features that are added to the original mask when OPC is utilized are typically sub-lithographic (i.e. have dimensions less than the resolution of the exposure tool) and thus do not transfer to the resist layer. Instead, they interact with the original pattern so as to improve the final transferred pattern and compensate for proximity effects. For instance, as shown in Figure 7, a desired pattern 710 may appear as actual pattern 720 when lithographically transferred without compensation for proximity effects. Using OPC techniques, positive serifs 732 and negative seriffs 734 may be added to the desired pattern 710 to form the mask 730 needed to compensate for proximity effects. Similarly, in Figure 7, the effects of proximity distortions on a typical desired transistor gate pattern 740 are shown by actual transferred pattern 750 and 752. When OPC corrections represented by hammerheads 762, assist bars 764 and bias bars 766 are added to the original desired mask pattern, the original desired shape will be more accurately transferred. In the case of transistor gates, the hammerhead shapes 762 are designed to eliminate the effect of line end shortening to ensure that the polysilicon portion of the gate extends beyond the active region 742. The assist bars 764 are designed to compensate for the isolated gate effect which tends to decrease the width of the transferred gate pattern. Finally, the bias bars 766 are designed to eliminate the effect of densely packed gates which is shown by the additional transferred pattern 752. In some instances, currently existing OPC products utilize a rule-based algorithm to generate proximity corrections for a given geometry. In this type of system, the design layout is analyzed for predetermined layout patterns and one of the aforementioned types of OPC features are generated for that area of the design layout. However, unlike one embodiment of the present invention, previous OPC products are not capable of retaining the true hierarchical data structure of the original design layout.

An embodiment of the present invention which is capable of providing for the generation of OPC corrections for an IC design layout while retaining the true hierarchical data structure of the original design layout is described below with respect to Fig. 8. This description includes by reference the above discussions of Figs. 2 and 4, as the system of Fig. 8 is a specific embodiment of the system and method described in these Figs. respectively.

Referring to Fig. 8, an integrated circuit chip design 800 is represented by hierarchical design data 810, which in one embodiment is in a GDS-II data format. The design data 810 is provided as an input to a computer system running an OPC algorithm 840 which incorporates one embodiment of the present invention. The computer system 840 operates to produce hierarchical correction data 845 in the manner described previously with respect to Figs. 2 and 4. In this respect, it can be seen that the computer system 840 comprises both the hierarchy preserver 210 and the operation engine 240 of Fig. 2, where the operation engine 240 of computer system 840 is a specifically defined OPC operation engine 240 that operates on the input design data 810 to provide for optical proximity corrections.

As shown in Fig. 8, the output hierarchical correction data 845 can be provided to a conventional design rule checker 850 along with the original design data 810 to design rule check the OPC corrected design. Similarly, the output could be used in the production of a lithography mask by combining the design data 810 with the correction data 845 as shown at block 860. This combined data can then be flattened and written to an EB machine as shown at block 865 in order for the EB machine to operate to produce the mask 870.

In one embodiment of the system of Fig. 8, the computer system 840 executes a computer program code stored on computer readable media that performs the functions of the compiler 220, the linker 230, and the OPC operation engine 240. In another embodiment, the computer system 840 comprises either a single computer system executing two or more different program codes or multiple separate computer systems executing two or more different program codes, one program code for the functions of the hierarchy preserver 210, and a separate program code for the functions of the OPC operation engine 240. In this embodiment, the hierarchy preserver 210 may selectively provide data to the OPC operation engine 240 through an API. With this embodiment, the hierarchy preserver 210 of the present invention can be modified to communicate and operate with currently existing OPC operation engines 240 to provide the advantages of hierarchical data output.

The computer readable media referred to above may comprise any computer storage tools including but not limited to hard disks, CDs, floppy disks, and server memory. The computer systems executing the program code may, in the case of both the OPC operation engine 240 and the hierarchy preserver 210, comprise any suitable computer system including a desktop computer running a Windows NT operating system or a Sun Solaris workstation for instance.

Operation engines which simply provide for OPC correction given a hierarchical input are well known in the field. In one embodiment of the system of Fig. 8, the OPC engine 240 is a rule-based OPC engine capable of generating OPC features in a manner that is controllable by the user of the system. For instance, the user can define the correction rules to be applied, and the size of the features to be applied to the design layout. Further, in one embodiment of the system, the location and size of bias lines 766 can be dependent upon the size and pitch of the IC pattern features being corrected, and/or restricted to being used only in critical areas of the design such as transistor gate regions. Still further, in another embodiment of the system the OPC engine 240 is capable of applying assist features 764 either in a localized manner to critical areas such as transistor gates or to the entire IC design globally. Still further, in another embodiment, the OPC engine can selectively place correction features in critical areas while not placing those correction features in areas that do not require them for accurate circuit performance. In one instance of this embodiment, the OPC engine can restrict the placing of biasing and assist features to transistor gates, while leaving the non-critical interconnect regions of the polysilicon gate layer uncorrected. In another instance, the OPC engine distinguishes critical transistor gate line-ends and applies hammerhead corrections to these areas to alleviate line-end shortening. Lastly, in another embodiment of the invention, the OPC operation engine is capable of providing for OPC correction of Phase Shifting Masks such as those disclosed in the United States patent application entitled, "Phase Shifting Circuit Manufacture Method and Apparatus" having serial number 08/931,921, filed September 17, 1997, and invented by Fang-Cheng Chang,Yao-Ting Wang and Yagyensh C. Pati which was previously incorporated by reference herein.

Fig. 9 illustrates how one embodiment of the system of Fig. 8 provides for OPC correction of the primitive geometries of leaf cells J1 and K1 of Fig. 1. Shown are uncorrected leaf cells J1 and K1 of Parent cell E1. The flattened primitive geometry data of J1 is provided to the hierarchy preserver 210, and the compiler 220 operates with the OPC engine 240 to provide a correction plane ΔJ1 in the manner discussed previously with respect to Fig. 2. In this case, the OPC engine has decided based on its rule definitions that the primitive geometry of J1 requires positive serifs 905 in order to provide the proper result when the mask is produced and used to expose a wafer. The same process is performed on the flattened primitive geometry of K1 to generate correction plane ΔK1, again comprising positive serifs 905. Each of these cells is then linked by the linker 230 as discussed previously to generate delta planes for each cell. Since these cells are leaf nodes and have no overlap areas, their respective delta planes are equal to their compiled correction planes. Also illustrated are corrected leaf cells 910 and 920 which represent J1 + ΔJ1, and K1 + ΔK1 respectively.

Figs. 10(a)-(b) illustrate how the method of Fig. 4 would generate an intermediate correction layer for overlap areas within Parent cell E1 of Fig. 1 for an OPC operation according to one embodiment of the invention. Fig. 10(b) illustrates the overlap area 1000 between corrected leaf cell J1 910 and the corrected leaf cell K1 920. As discussed above with respect to Figs. 2 and 4, during the linking process for cell E1, this overlap area is determined and the data corresponding to this area is flattened. The flattened overlap area is then provided to the OPC operation engine 240 which operates on the data to provide an intermediate correction plane 1020. Note that in the case described here where the primitive structures overlap a discrete amount, negative serifs 1010 are provided for the intermediate correction plane. In the situation described below with respect to Fig. 10(b), an alternate parent E1 is illustrated in which the corrected leaf cells K1 and K2 shown as 910b and 920b respectively. This situation illustrates an infinitesimal overlap between the two corrected primitive geometries. In one embodiment of the invention, an intermediate correction plane 1020b is provided for these infinitesimal overlap situations such that a -2 layer is provided to compensate for end butting effects.

Fig. 11 illustrates how the method of Fig. 4 would generate the delta plane of parent cell E1 of Fig. 1 for an OPC operation according to one embodiment of the invention. As described by block 1100, at the link stage for cell E1, the overlap areas within E1 are determined and the area data is flattened. This is illustrated as overlap area 1000. Next, as described by block 1110, an intermediate correction plane 1020 for this overlap area 1000 is generated as described above with respect to Fig. 10(a). At block 1120, the correction planes 910 and 920 of all the children cells of E1 are summed to generate summed children correction data 1140. The last step as described by block 1130 is to generate the delta plane 1150 for cell E1 and store this data hierarchically. This is accomplished in one embodiment by subtracting the summed children correction data 1140 from the intermediate correction plane 1020 to obtain the delta plane 1150. Also illustrated in Fig. 11 is the final correction plane 1160 for cell E1 which as defined earlier represents the total of the corrections needed to be applied to the cell's design data in order to properly apply the particular operation, which here is OPC, to the cell. The correction plane 1160 comprises the E1's delta plane summed with the correction planes 910 and 920 of its children J1 and K1 respectively.

Fig. 12 illustrates a further method for providing OPC correction to a design layout using one embodiment of the present invention. At block 1200 an integrated circuit design layout is first provided. The hierarchically formatted design data corresponding to this design layout is then provided to a system which performs a rule based OPC correction on the design data in accordance with the system of Fig. 8 as shown at block 1205. The system of Fig. 8 generates an output of hierarchical correction data as described above, and this correction data is combined with the original design data to generate a hierarchically described rule-based OPC corrected design data as shown at block 1210. Using this corrected design data a simulated image of the mask which this corrected design data would produce is generated at block 1215. This simulation can be generated utilizing a Hopkins equation based simulation device such as that described generally in the United States provisional patent application entitled, "Mask Verification, Correction, and Design Rule Checking" having serial number 60/059,306, filed September 17, 1997, and invented by Fang-Cheng Chang, Yao-Ting Wang and Yagyensh C. Pati, United States patent application entitled, "Mask Verification, Correction, and Design Rule Checking", filed September 16, 1998, and invented by Fang-Cheng Chang, Yao-Ting Wang and Yagyensh C. Pati, and also more specifically in the United States patent application entitled "Visual Inspection and Verification System", filed August 7, 1998, and invented by Fang-Cheng Chang, Yao-Ting Wang, Yagyensh C. Pati, and Linard Karklin, each of which were previously incorporated by reference herein.

The simulated image of the corrected mask is then compared to the desired design image at block 1220 to determine whether or not the initial rule based OPC correction was sufficient to correct the design to within a set of user defined design parameters as shown at block 1225. Methods for performing this comparison are disclosed in the aforementioned United States provisional patent application entitled, "Mask Verification, Correction, and Design Rule Checking", and the United States utility application of the same name. If the result of the comparison is that the design parameters have been achieved, the corrected design data can then be input to a design rule checker which analyzes the corrected design for any violations of the established design rules for the particular integrated circuit design as shown at block 1235. If the corrected design is within design rules, the corrected design data can then be flattened and a mask can be produced using an EB machine as shown at block 1245. If the design rules have not been met, a decision must be made as to whether or not to redesign the mask as shown at block 1250.

If the decision is not to redesign, but to try and fix the problem by continuing with the iterative correction process, a model based OPC algorithm is then run on the corrected design. Similarly, if the original corrected design data did not meet the design parameters of block 1225, the original corrected design data is input to a model based OPC algorithm. The model based OPC algorithm is then used to perform more detailed specific corrections to the original corrected design as shown at block 1230. The model based OPC corrected design can then be provided to block 1215 where a simulated image of the model based OPC corrected design is produced and once again compared to the desired image. Before entering the OPC corrected design into a conventional design rule checker product for analysis of the design, the simulated image of the model based OPC corrected design must be manipulated into a format that is acceptable by a conventional design rule checker. One way of doing this is to generate a Manhattan geometry representation of the simulated image based on an edge checking technique as described more fully in the aforementioned and incorporated United States provisional patent application entitled, "Mask Verification, Correction, and Design Rule Checking" and the United States utility patent application of the same name. This whole process can be continued until a corrected design is produced which meets both the user defined design parameters and the circuit specific design rules.

In one embodiment of this process, the model based OPC algorithm is capable of responding to user defined input. For instance, in one embodiment, the user can control the complexity level of the corrections he wants to be applied in order to control data volume and overall process speed. Similarly, in another embodiment the user can control the size of correction features to be applied by the model based algorithm. Still further, in another embodiment, the user can define the correction criteria to be applied by the algorithm.

The remaining Figs. 13-19 illustrate example screen snapshots from a computer system executing one embodiment of the present invention to provide OPC corrections for a hierarchical input IC design layout. For instance, Fig. 13 illustrates an example screen snapshot of the input design layout which is to be OPC corrected. The user interface 1300 of the design program comprises a design window 1330 in which is illustrated a portion of the IC design layout to be corrected. The design layout includes a diffusion layer 1390 and a layer of polysilicon structures such as primitive structures 1320. A cell 1310, similar to the sample parent cells E1 and F1 of Fig. 1, is also depicted in design window 1330.

Fig. 14 illustrates an example screen snapshot of the final output from a computer system executing one embodiment of the invention to provide OPC correction to the input design of Fig. 13. The design window 1330 of user interface 1300 shows a cell 1310 comprising primitive structures 1320 which have been OPC corrected. The cell 1310 comprises OPC features such as hammerheads 1410, assist lines 1420, bias lines 1430, positive serifs 1440, and negative serifs 1450. The output depicted in Fig. 14 is representative of all of the corrections which would have to be made to compensate for all OPC effects on the entire design. Thus, these corrections represent the final linked output of this embodiment of the present invention in which all overlaps between cells in the hierarchy have been resolved and compensated for. The OPC features depicted in Fig. 14 are shown in greater detail in Fig. 15 which is a zoomed in example screen snapshot of Fig. 14.

Fig. 16 illustrates an example screen snapshot of a -1 OPC correction layer from a computer system executing one embodiment of the invention to provide OPC correction. This layer contains corrections to cell 1310 including assist lines 1420, bias lines 1430, and negative serifs 1450.

Fig. 17 illustrates an example screen snapshot of a +1 OPC correction layer from a computer system executing one embodiment of the invention to provide OPC correction. This layer contains corrections to cell 1310 including hammerheads 1410, assist lines 1420, and positive serifs 1440.

Fig. 18 illustrates an example screen snapshot of a -2 OPC correction layer from a computer system executing one embodiment of the invention to provide OPC correction. This layer contains corrections to cell 1310 including end butting correction feature 1810.

Fig. 19 illustrates an example screen snapshot of an individual cell 1310 that has been OPC corrected by a computer system executing one embodiment of the invention. The design window 1330 illustrates a cell 1310 to which its linked correction layer has been applied. The corrections applied to the cell 1310 include hammerheads 1410, assist lines 1420, positive serifs 1440, and negative serifs 1450. Note that the corrections to cell 1310 are different than those illustrated in Fig. 14 because Fig. 14 is a representation of all the corrections to the entire design -- while Fig. 19 only illustrates those corrections necessary to correct cell 1310 individually. In other words, the corrections illustrated in Fig. 19 do not take into account the interactions of cell 1310 with other cells adjacent to it. For instance, note that the bias lines 1430 of Fig. 14 are not present in Fig. 19.

Although illustrative embodiments of the invention have been described in detail herein with reference to the accompanying drawings, it is to be understood that the invention is not limited to those precise embodiments. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed. As such, many modifications and variations will be apparent to practitioners skilled in this art. Accordingly, it is intended that the scope of the invention be defined by the following claims and their equivalents.

6. The Claims

What is claimed is:

1. A method of generating optical proximity corrections for an integrated circuit layout, wherein the data describing the integrated circuit layout comprises a hierarchical structure, the method comprising:

providing the integrated circuit layout design as a first input;

providing a particular set of correction criteria as a second input;

analyzing the integrated circuit layout to identify features of the layout that meet the particular set of correction criteria;

generating optical proximity correction data in response to the particular set of correction criteria for the features that meet the particular set of correction criteria; and

providing a first program data wherein the first program data comprises the optical proximity correction data configured in a hierarchical structure that corresponds to the hierarchical structure of the integrated circuit layout.

2. The method of generating optical proximity corrections for an integrated circuit layout of claim 1 wherein the first program data is provided on a computer readable media.

3. The method of generating optical proximity corrections for an integrated circuit layout of claim 2 wherein the computer readable media comprises a hard disk drive.

4. The method of generating optical proximity corrections for an integrated circuit layout of claim 2 wherein the computer readable media comprises a server.

5. The method of generating optical proximity corrections for an

integrated circuit layout of claim 1 wherein the integrated circuit layout is described by a GDS-II data file.

6. The method of generating optical proximity corrections for an integrated circuit layout of claim 5 wherein the first program data is described by a GDS-II data file.

7. The method of generating optical proximity corrections for an integrated circuit layout of claim 1 wherein the optical proximity correction data comprises data corresponding to the addition of serifs to the layout.

8. The method of generating optical proximity corrections for an integrated circuit layout of claim 1 wherein the optical proximity correction data comprises data corresponding to the correction of transistor gates.

9. The method of generating optical proximity corrections for an integrated circuit layout of claim 1 wherein the particular set of correction criteria comprises a criteria for specifically identifying transistor gates in the layout.

10. The method of generating optical proximity corrections for an integrated circuit layout of claim 9 wherein the optical proximity correction data comprises data corresponding to the addition of hammerhead features to transistor line ends.

11. The method of generating optical proximity corrections for an integrated circuit layout of claim 1 further comprising combining the first program data with the data describing the integrated circuit layout to produce a second program data that describes a first corrected layout.

12. The method of generating optical proximity corrections for an integrated circuit layout of claim 11 further comprising flattening the second program data to produce a third program data, and utilizing the third program data to produce an optically corrected lithographic mask.

13. The method of generating optical proximity corrections for an integrated circuit layout of claim 11 further comprising providing the second program data to a design rule checker to determine whether the first corrected layout falls within a set of design rules associated with the integrated circuit.

14. The method of generating optical proximity corrections for an integrated circuit layout of claim 11 further comprising:

providing a set of layout accuracy parameters;

comparing the first corrected layout to the design accuracy parameters;

providing a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules; and

applying the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within the set of layout accuracy parameters.

15. The method of generating optical proximity corrections for an integrated circuit layout of claim 14 further comprising:

providing the second corrected layout to a design rule checker;

operating the design rule checker; and

determining whether the second corrected layout falls within a set of design rules associated with the integrated circuit.

16. The method of generating optical proximity corrections for an integrated circuit layout of claim 14, wherein comparing the first corrected layout to the design accuracy parameters further comprises:

providing the second program data to a checker device which produces a simulated image of the exposure that the first corrected layout would produce;

providing the actual image of the exposure that was designed for the integrated circuit; and

measuring the difference between the simulated image and the actual image.

17. The method of generating optical proximity corrections for an integrated circuit layout of claim 1 wherein the integrated circuit layout comprises a plurality of cells in a tree structure, and wherein providing the first program data comprises:

generating a plurality of delta planes corresponding to the plurality of cells wherein each delta plane comprises data representative of the difference between a correction plane of the cell corresponding to the delta plane and the delta planes corresponding to the children cells of the cell corresponding to the delta plane.

18. The method of generating optical proximity corrections for an integrated circuit layout of claim 17 wherein the correction plane for each cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data.

19. A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout, wherein the data describing the integrated circuit layout comprises a hierarchical structure, the method comprising:

providing the integrated circuit layout as a first input;

providing a particular set of correction criteria as a second input;

analyzing the integrated circuit layout to identify features of the layout that meet the particular set of correction criteria;

generating optical proximity correction data in response to the particular set of correction criteria for the features that meet the particular set of correction rules; and

providing a first program data wherein the first program data comprises the optical proximity correction data configured in a hierarchical structure that corresponds to the hierarchical structure of the integrated circuit layout.

20. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 19 wherein the first program data is provided on a computer readable media.

21. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 20 wherein the computer readable media comprises a hard disk drive.

22. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 20 wherein the computer readable media comprises a server.

23. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 19 wherein the integrated circuit layout is described by a

GDS-II data file.

24. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 23 wherein the first program data is described by a GDS-II data file.

25. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 19 wherein the optical proximity correction data comprises data corresponding to the addition of serifs to the layout.

26. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 19 wherein the optical proximity correction data comprises data corresponding to the correction of transistor gates.

27. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 19 wherein the particular set of correction criteria comprises a means for specifically identifying transistor gates in the layout.

28. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 27 wherein the optical proximity correction data

comprises data corresponding to the addition of hammerhead features to transistor line ends.

29. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 19, the method further comprising combining the first program data with the data describing the integrated circuit mask to produce a second program data that describes a first corrected layout.

30. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 29, the method further comprising flattening the second program data to produce a third program data, and utilizing the third program data to produce an optically corrected lithographic mask.

31. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 29, the method further comprising providing the second program data to a design rule checker to determine whether the first corrected layout falls within a set of design rules associated with the integrated circuit.

32. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 29, the method further comprising:

providing a set of layout accuracy parameters;

comparing the first corrected layout to the design accuracy parameters;

providing a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules; and

applying the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within thc set of layout accuracy parameters.

33. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 32, the method further comprising:

providing the second corrected layout to a design rule checker;

operating the design rule checker; and

determining whether the second corrected layout falls within a set of design rules associated with the integrated circuit.

34. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 32, wherein comparing the first corrected layout to the design accuracy parameters further comprises:

providing the second program data to a checker device which produces a simulated image of the exposure that the first corrected layout would produce;

providing the actual image of the exposure that was designed for the integrated circuit; and

measuring the difference between the simulated image and the actual image.

35. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated

circuit layout of claim 19 wherein the integrated circuit layout comprises a plurality of cells in a tree structure, and wherein providing the first program data comprises:

generating a plurality of delta planes corresponding to the plurality of cells wherein each delta plane comprises data representative of the difference between a correction plane of the cell corresponding to the delta plane and the delta planes corresponding to the children cells of the cell corresponding to the delta plane.

36. The program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for generating optical proximity corrections for an integrated circuit layout of claim 35 wherein the correction plane for each cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data.

37. An apparatus for generating optical proximity corrections for an integrated circuit layout, wherein the data describing the integrated circuit layout comprises a hierarchical structure, the apparatus comprising:

a first input that receives the integrated circuit layout;

a second input that receives a particular set of correction criteria;

a resource that analyzes the integrated circuit layout and identifies features that meet the particular set of correction criteria;

a resource that generates optical proximity correction data in response to the particular set of correction criteria for the features that meet the particular set of correction criteria; and

a resource that provides a first program data wherein the first program data comprises the optical proximity correction data configured in a hierarchical structure that corresponds to the hierarchical structure of the integrated circuit layout.

38. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 37 wherein the first program data is provided on a computer readable media.

39. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 38 wherein the computer readable media comprises a hard disk drive.

40. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 38 wherein the computer readable media comprises a server.

41. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 37 wherein the integrated circuit layout is described by a GDS-II data file.

42. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 41 wherein the first program data is described by a GDS-II data file.

43. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 37 wherein the optical proximity correction data comprises data corresponding to the addition of serifs to the layout.

44. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 37 wherein the optical proximity correction data comprises data corresponding to the correction of transistor gates.

45. The apparatus for generating optical proximity corrections for an

integrated circuit layout of claim 37 wherein the particular set of correction criteria comprises a means for specifically identifying transistor gates in the layout.

46. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 45 wherein the optical proximity correction data comprises data corresponding to the addition of hammerhead features to transistor line ends.

47. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 37 further comprising a resource that combines the first program data with the data describing the integrated circuit mask to produce a second program data that describes a first corrected layout.

48. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 47 further comprising a resource that flattens the second program data to produce a third program data, and a resource that utilizes the third program data to produce an optically corrected lithographic mask.

49. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 47 further comprising a design rule checker that receives the second program data and determines whether the first corrected layout falls within a set of design rules associated with the integrated circuit.

50. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 47 further comprising:

a third input that receives a set of layout accuracy parameters;

a resource that compares the first corrected layout to the design accuracy parameters;

a resource that provides a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules; and

a resource that applies the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within the set of layout accuracy parameters.

51. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 50 further comprising a design rule checker that receives the second corrected mask and determines whether the second corrected layout falls within a set of design rules associated with the integrated circuit.

52. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 50, wherein comparing the first corrected layout to the design accuracy parameters further comprises:

a resource that provides the second program data to a checker device which produces a simulated image of the exposure that the first corrected layout would produce;

a resource that provides the actual image of the exposure that was designed for the integrated circuit; and

a resource that measures the difference between the simulated image and the actual image.

53. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 37 wherein the integrated circuit layout comprises a plurality of cells in a tree structure, and wherein the resource for providing the first program data further:

generates a plurality of delta planes corresponding to the plurality of cells wherein each delta plane comprises data representative of the difference between a correction plane of the cell corresponding to the delta plane and the delta planes corresponding to the children cells of the cell corresponding to the delta plane.

54. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 53 wherein the correction plane for each cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data.

55. A computer program product, comprising:

a computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout, wherein the data describing the integrated circuit layout comprises a hierarchical structure, the computer readable program code comprising:

computer readable program code that reads the integrated circuit layout as a first input;

computer readable program code that reads a particular set of correction criteria as a second input;

computer readable program code that analyzes the integrated circuit layout to identify features that meet the particular set of correction criteria;

computer readable program code that generates optical proximity correction data in response to the particular set of correction criteria for the features that meet the particular set of correction criteria; and

computer readable program code that provides a first program data wherein the first program data comprises the optical proximity correction data configured in a hierarchical structure that corresponds to the hierarchical structure of the integrated circuit layout.

56. The computer usable medium having a computer readable

program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 55 wherein the first program data is provided on a computer readable media.

57. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 56 wherein the computer readable media comprises a hard disk drive.

58. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 56 wherein the computer readable media comprises a server.

59. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 55 wherein the integrated circuit layout is described by a GDS-II data file.

60. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 59 wherein the first program data is described by a GDS-II data file.

61. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 55 wherein the optical proximity correction data comprises data corresponding to the addition of serifs to the layout.

62. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 55 wherein the optical proximity correction data comprises data corresponding to the correction of transistor gates.

63. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 55 wherein the particular set of correction criteria comprises a means for specifically identifying transistor gates in the layout.

64. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 63 wherein the optical proximity correction data comprises data corresponding to the addition of hammerhead features to transistor line ends.

65. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 55 further comprising computer readable program code that combines the first program data with data describing the integrated circuit mask to produce a second program data that describes a first corrected layout.

66. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 65 further comprising a computer readable program code that flattens the second program data to produce a third program data, and a computer readable program code

that utilizes the third program data to produce an optically corrected lithographic mask.

67. The computer usablc medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 65 further comprising a computer readable program code that receives the second program data as a third input and determines whether the first corrected layout falls within a set of design rules associated with the integrated circuit.

68. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 65 further comprising:

a computer readable program code that receives a set of layout accuracy parameters as a third input;

a computer readable program code that compares the first corrected layout to the design accuracy parameters;

a computer readable program code that provides a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules; and

a computer readable program code that applies the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within the set of layout accuracy parameters.

69. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 68 further comprising a computer readable program code that receives the second

corrected mask as a fourth input and determines whether the second corrected layout falls within a set of design rules associated with the integrated circuit.

70. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 68, wherein the computer readable program code that compares the optically corrected layout to the design accuracy parameters further comprises:

a computer readable program code that receives the second program and produces a simulated image of the exposure that the optically corrected layout would produce;

a computer readable program code that provides the actual image of the exposure that was designed for the integrated circuit; and

a computer readable program code that measures the difference between the simulated image and the actual image.

71. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 55, wherein the computer readable program code that provides a first program data computer further comprises:

computer readable program code that generates a plurality of delta planes corresponding to the plurality of cells wherein each delta plane comprises data representative of the difference between a correction plane of the cell corresponding to the delta plane and the delta planes corresponding to the children cells of the cell corresponding to the delta plane.

72. The computer usable medium having a computer readable program code embodied therein for causing a computer to generate optical proximity corrections for an integrated circuit layout of claim 71 wherein the

correction plane for each cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data.

73. A method of generating optical proximity corrections for an integrated circuit layout, wherein the data describing the integrated circuit layout comprises a hierarchical tree structure comprising a plurality of cells, the method comprising:

providing the integrated circuit layout as a first input;

providing a particular set of correction criteria as a second input;

compiling the hierarchical tree structure, wherein compiling comprises generating a first correction layer for each cell of the plurality of cells in the hierarchical tree structure in response to the particular set of correction criteria;

linking the hierarchical tree structure, wherein linking comprises modifying the correction layer of each cell to generate a delta plane for each cell such that the delta plane of each cell accounts for interaction between each of the cell's child cells and interaction between the cell's primitive geometry and each of the cell's child cells; and

providing a first program data comprising the delta planes, wherein the first program data is configured hierarchically such that it corresponds to the hierarchical tree structure of the integrated circuit layout.

74. The method of generating optical proximity corrections for an integrated circuit layout of claim 73 wherein for each cell in the hierarchical tree structure the sum of the delta plane of the cell and the delta planes of the cell's child cells comprises a correction plane of the cell wherein the correction plane for the cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data.

75. The method of generating optical proximity corrections for an integrated circuit layout of claim 73 wherein the first program data is provided on a computer readable media.

76. The method of generating optical proximity corrections for an integrated circuit layout of claim 73 wherein the integrated circuit layout is described by a GDS-II data file.

77. The method of generating optical proximity corrections for an integrated circuit layout of claim 76 wherein the first program data is described by a GDS-II data file.

78. The method of generating optical proximity corrections for an integrated circuit layout of claim 73 wherein the particular set of correction criteria comprises a means for specifically identifying transistor gates in the layout.

79. The method of generating optical proximity corrections for an integrated circuit layout of claim 73 further comprising combining the first program data with the data describing the integrated circuit layout to produce a second program data that describes a first corrected layout.

80. The method of generating optical proximity corrections for an integrated circuit layout of claim 79 further comprising flattening the second program data to produce a third program data, and utilizing the third program data to produce an optically corrected lithographic mask.

81. The method of generating optical proximity corrections for an integrated circuit layout of claim 79 further comprising providing the second program data to a design rule checker to determine whether the first corrected

layout falls within a set of design rules associated with the integrated circuit.

82. The method of generating optical proximity corrections for an integrated circuit layout of claim 79 further comprising:

providing a set of layout accuracy parameters;

comparing the first corrected layout to the design accuracy parameters;

providing a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules; and

applying the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within the set of layout accuracy parameters.

83. The method of generating optical proximity corrections for an integrated circuit layout of claim 82 further comprising providing the second corrected layout to a design rule checker to determine whether the second corrected layout falls within a set of design rules associated with the integrated circuit.

84. An apparatus for generating optical proximity corrections for an integrated circuit layout, wherein the data describing the integrated circuit layout comprises a hierarchical tree structure comprising a plurality of cells, the apparatus comprising:

a first input that receives the integrated circuit layout;

a second input that receives a particular set of correction criteria;

a resource that compiles the hierarchical tree structure, wherein compiling comprises generating a first correction layer for each cell of the plurality of cells in the tree structure in response to the particular set of correction criteria;

a resource that links the hierarchical tree structure, wherein linking comprises modifying the correction layer of each cell to generate a delta plane for each cell such that the delta plane of each cell accounts for interaction between each of the cell's child cells and interaction between the cell's primitive geometry and each of the cell's child cells; and

a resource that provides a first program data comprising the delta planes, wherein the first program data is configured hierarchically such that it corresponds to the hierarchical tree structure of the integrated circuit layout.

85. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 84 wherein for each cell in the hierarchical tree structure the sum of the delta plane of the cell and the delta planes of the cell's child cells comprises a correction plane of the cell wherein the correction plane for the cell comprises data that would generate an output data equal to a corrected design for the cell if the correction plane were applied to the flattened cell data.

86. The method of generating optical proximity corrections for an integrated circuit layout of claim 84 wherein the first program data is provided on a computer readable media.

87. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 84 wherein the integrated circuit layout is described by a GDS-II data file.

88. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 87 wherein the first program data is described by a GDS-II data file.

89. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 84 wherein the particular set of correction criteria comprises a means for specifically identifying transistor gates in the

layout.

90. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 84 further comprising a resource that combines the first program data with data describing the integrated circuit layout to produce a second program data that describes a first corrected layout.

91. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 90 further comprising a resource that flattens the second program data to produce a third program data, and a resource that utilizes the third program data to produce an optically corrected lithographic mask.

92. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 90 further comprising a design rule checker that receives the second program data and determines whether the first corrected layout falls within a set of design rules associated with the integrated circuit.

93. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 90 further comprising:

a third input that receives a set of layout accuracy parameters;

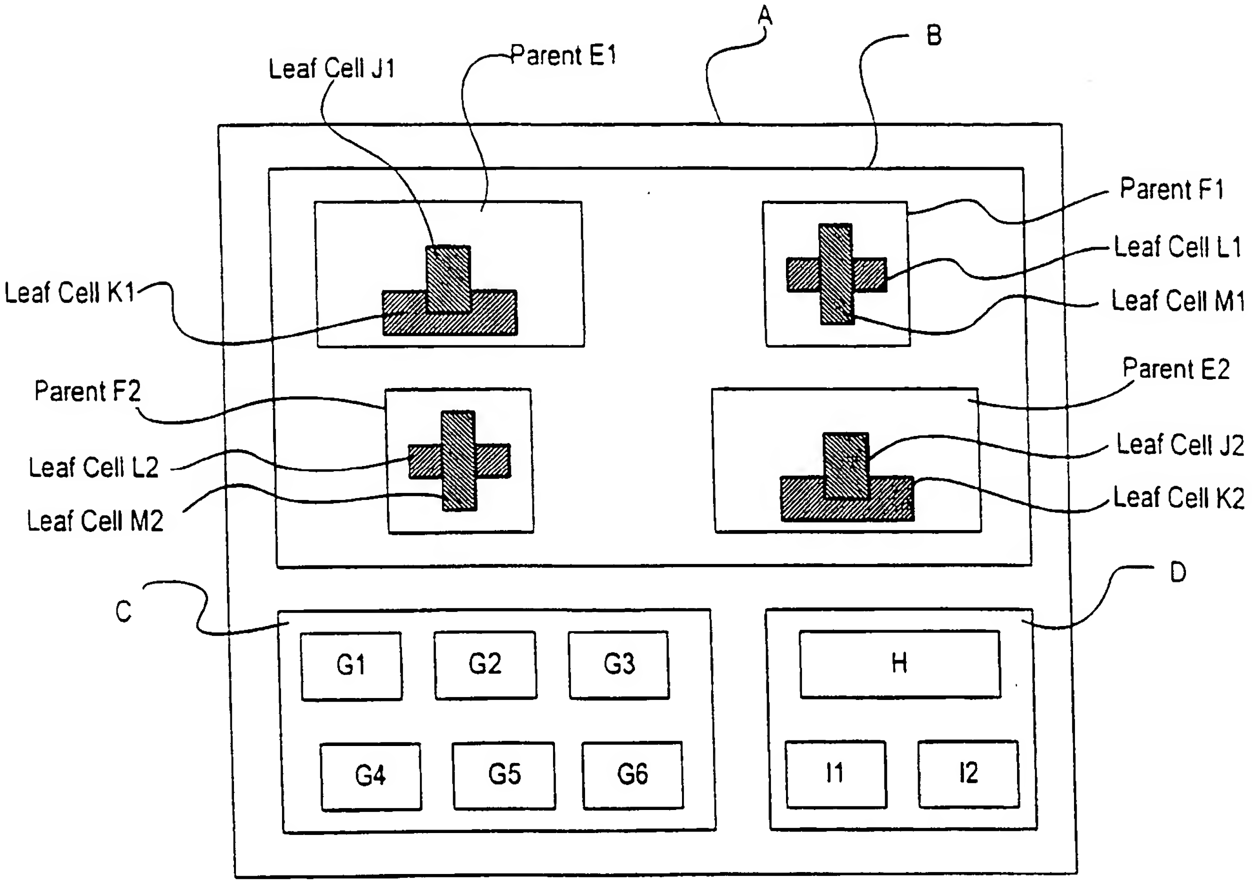
a resource that compares the first corrected layout to the design accuracy parameters;

a resource that provides a model based correction means for correcting all areas of the layout in accordance with a particular set of design accuracy rules; and

a resource that applies the model based correction means to the first corrected layout to produce a second corrected layout such that the second corrected layout falls within the set of layout accuracy parameters.

94. The apparatus for generating optical proximity corrections for an integrated circuit layout of claim 93 further comprising a resource that provides the second corrected layout to a design rule checker to determine whether the second corrected layout falls within a set of design rules associated with the integrated circuit.

100 - Simple Integrated Circuit Layout



110 - Hierarchical Representation of Layout

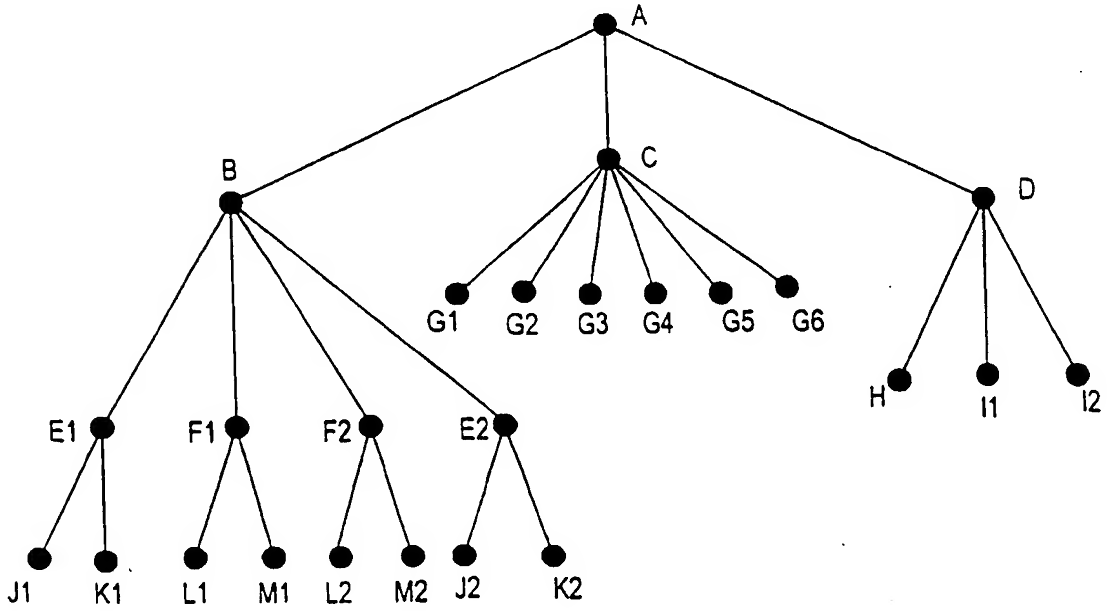


Figure 1 (Prior Art)

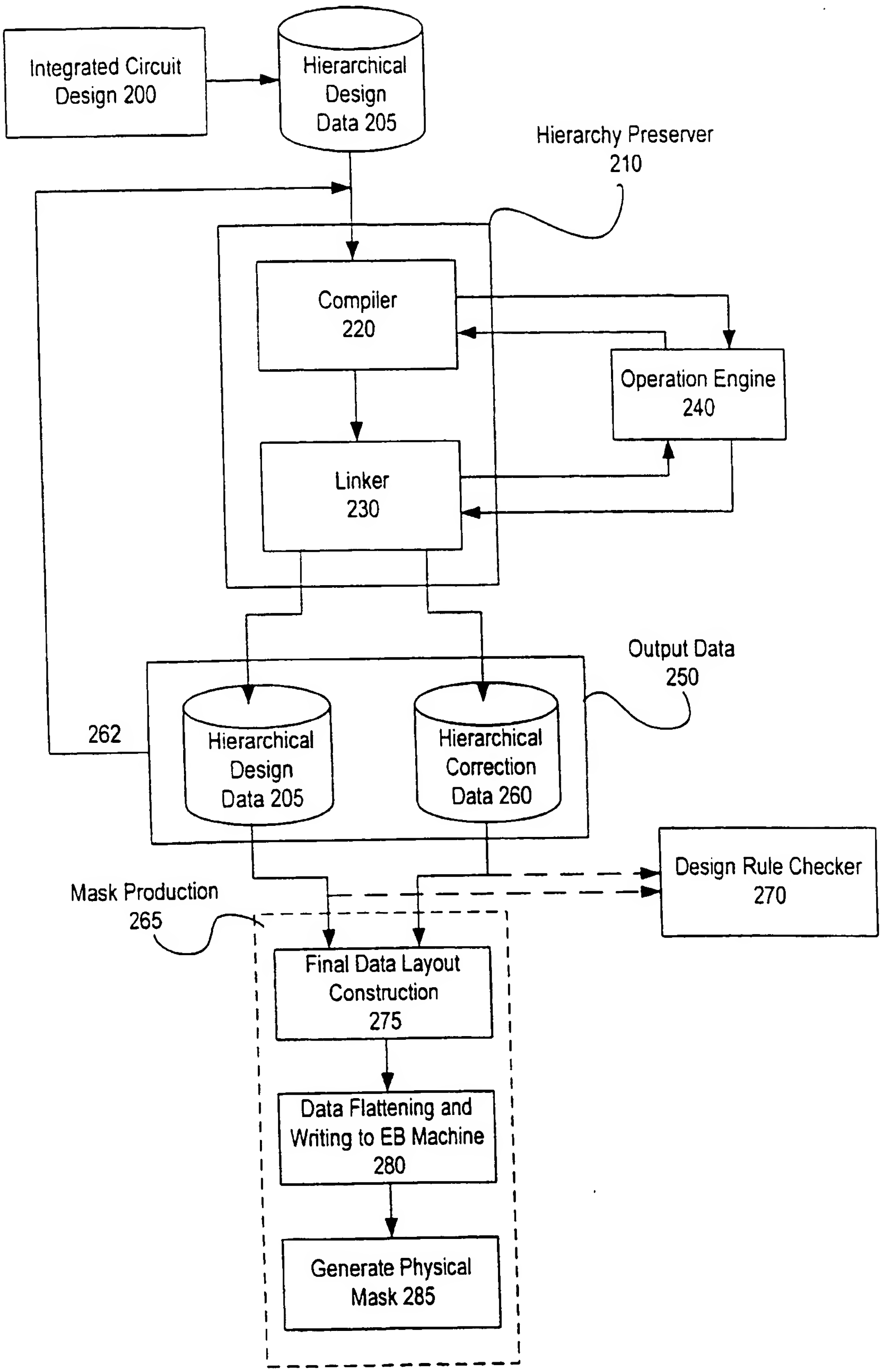

Integrated Circuit Design 200
Hierarchical Design Data 205
Hierarchy Preserver 210
Compiler 220
Operation Engine 240
Linker 230
Output Data 250
262
Hierarchical Design Data 205
Hierarchical Correction Data 260
Design Rule Checker 270
Mask Production 265
Final Data Layout Construction 275
Data Flattening and Writing to EB Machine 280
Generate Physical Mask 285

Figure 2

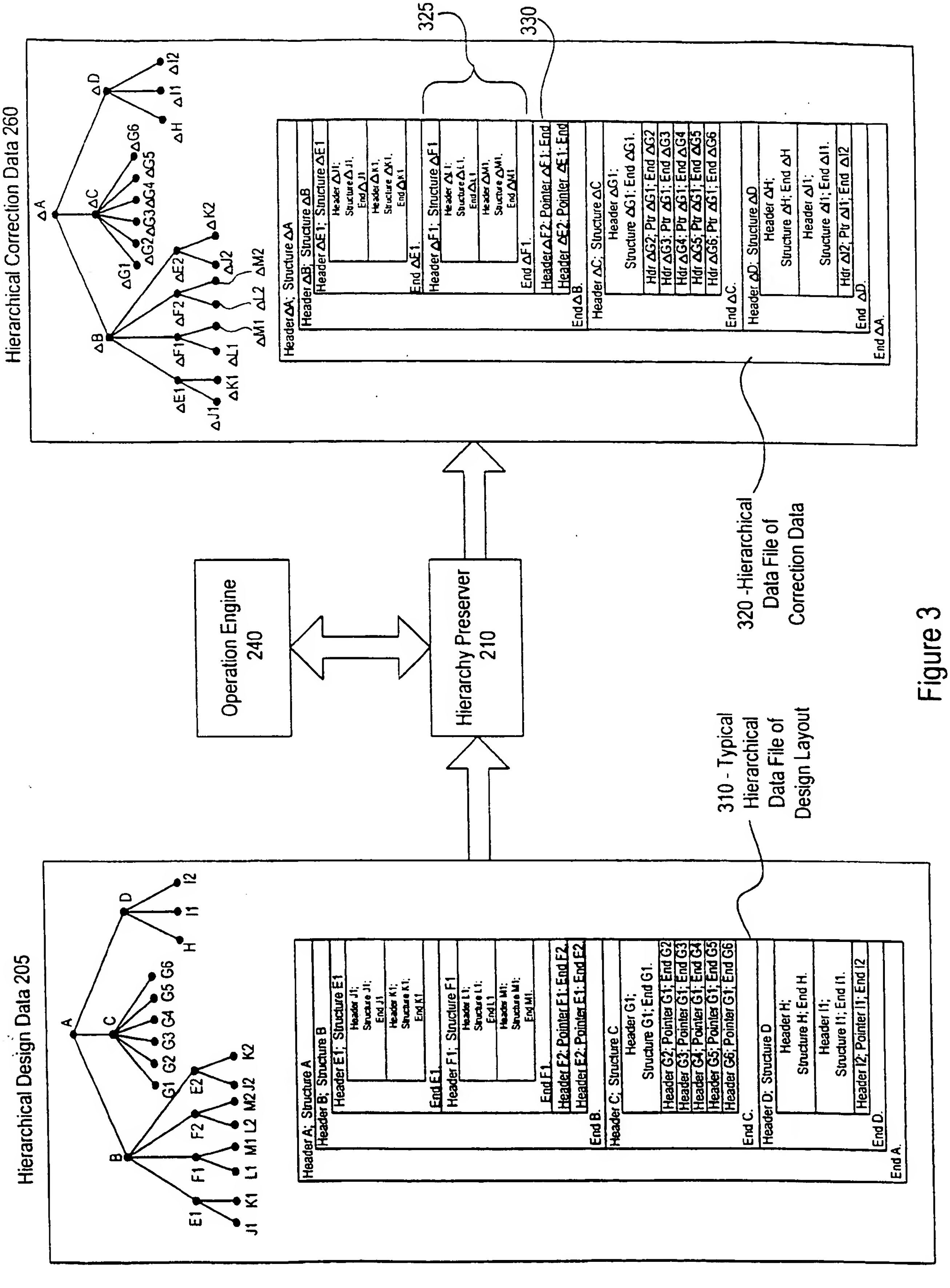

Hierarchical Design Data 205
A
B
C
D
E1
F1
F2
E2
G1 G2 G3 G4 G5 G6
H I1 I2
J1 K1 L1 M1 L2 M2 J2 K2
Header A; Structure A
Header B; Structure B
Header E1; Structure E1
Header J1; Structure J1; End J1
Header K1; Structure K1; End K1
End E1.
Header F1; Structure F1
Header L1; Structure L1; End L1
Header M1; Structure M1; End M1.
End F1.
Header F2; Pointer F1; End F2.
Header E2; Pointer E1; End E2.
End B.
Header C; Structure C
Header G1; Structure G1; End G1.
Header G2; Pointer G1; End G2
Header G3; Pointer G1; End G3
Header G4; Pointer G1; End G4
Header G5; Pointer G1; End G5
Header G6; Pointer G1; End G6
End C.
Header D; Structure D
Header H; Structure H; End H.
Header I1; Structure I1; End I1.
Header I2; Pointer I1; End I2
End D.
End A.
310 - Typical Hierarchical Data File of Design Layout
Operation Engine 240
Hierarchy Preserver 210
Hierarchical Correction Data 260
ΔA
ΔB
ΔC
ΔD
ΔE1
ΔF1
ΔF2
ΔE2
ΔG1 ΔG2 ΔG3 ΔG4 ΔG5 ΔG6
ΔH ΔI1 ΔI2
ΔJ1 ΔK1 ΔL1 ΔM1 ΔL2 ΔM2 ΔJ2 ΔK2
325
330
Header ΔA; Structure ΔA
Header ΔB; Structure ΔB
Header ΔE1; Structure ΔE1
Header ΔJ1; Structure ΔJ1; End ΔJ1
Header ΔK1; Structure ΔK1; End ΔK1
End ΔE1.
Header ΔF1; Structure ΔF1
Header ΔL1; Structure ΔL1; End ΔL1
Header ΔM1; Structure ΔM1; End ΔM1
End ΔF1.
Header ΔF2; Pointer ΔE1; End
Header ΔE2; Pointer ΔE1; End
End ΔB.
Header ΔC; Structure ΔC
Header ΔG1; Structure ΔG1; End ΔG1.
Hdr ΔG2; Ptr ΔG1; End ΔG2
Hdr ΔG3; Ptr ΔG1; End ΔG3
Hdr ΔG4; Ptr ΔG1; End ΔG4
Hdr ΔG5; Ptr ΔG1; End ΔG5
Hdr ΔG6; Ptr ΔG1; End ΔG6
End ΔC.
Header ΔD; Structure ΔD
Header ΔH; Structure ΔH; End ΔH
Header ΔI1; Structure ΔI1; End ΔI1.
Hdr ΔI2; Ptr ΔI1; End ΔI2
End ΔD.
End ΔA.
320 -Hierarchical Data File of Correction Data

Figure 3

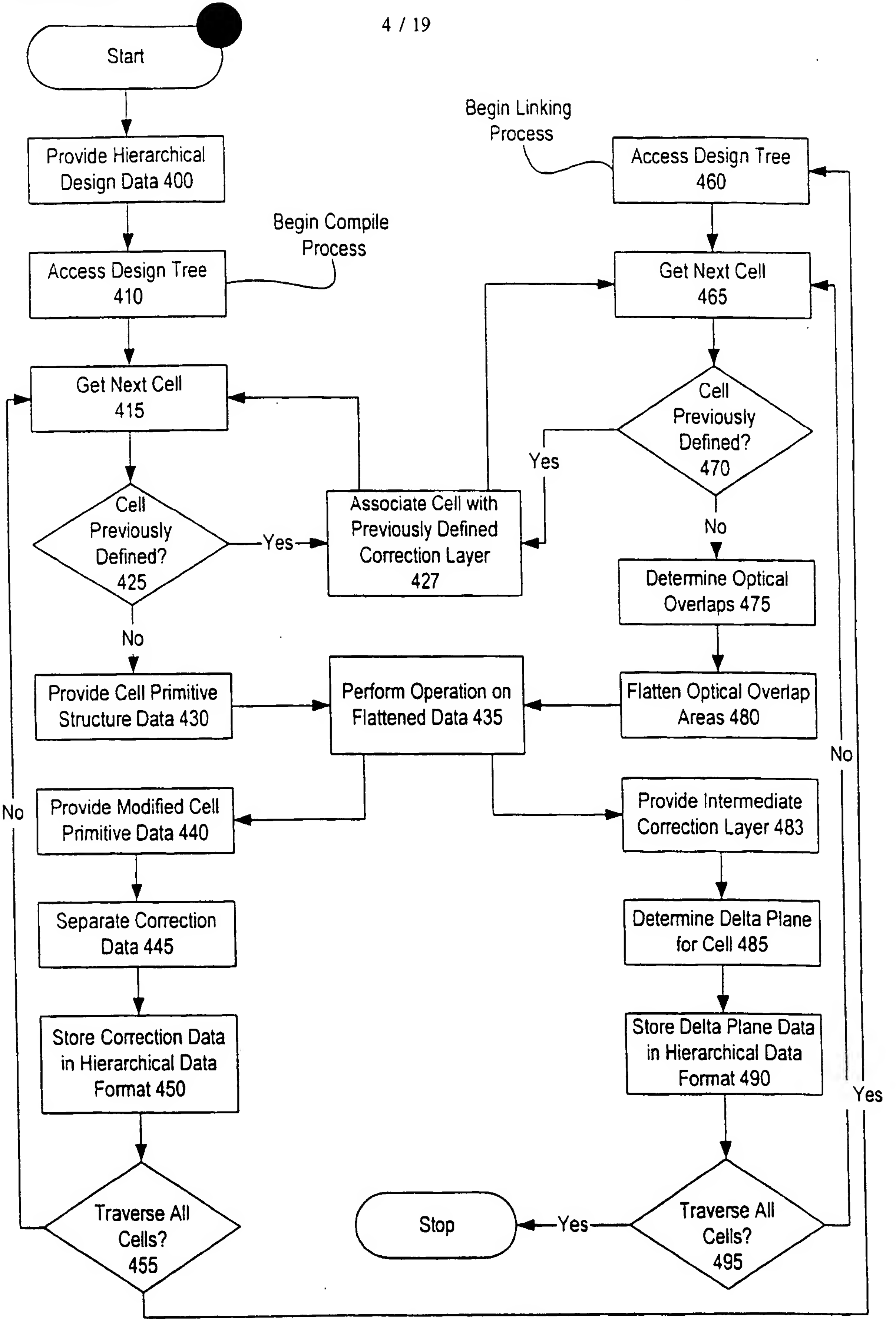

Start
Provide Hierarchical Design Data 400
Access Design Tree 410
Begin Compile Process
Get Next Cell 415
Cell Previously Defined? 425
Yes
Associate Cell with Previously Defined Correction Layer 427
No
Provide Cell Primitive Structure Data 430
Perform Operation on Flattened Data 435
Provide Modified Cell Primitive Data 440
Separate Correction Data 445
Store Correction Data in Hierarchical Data Format 450
Traverse All Cells? 455
No
Begin Linking Process
Access Design Tree 460
Get Next Cell 465
Cell Previously Defined? 470
Yes
No
Determine Optical Overlaps 475
Flatten Optical Overlap Areas 480
Provide Intermediate Correction Layer 483
Determine Delta Plane for Cell 485
Store Delta Plane Data in Hierarchical Data Format 490
Traverse All Cells? 495
No
Yes
Yes
Stop

Figure 4

Typical OPC Corrections

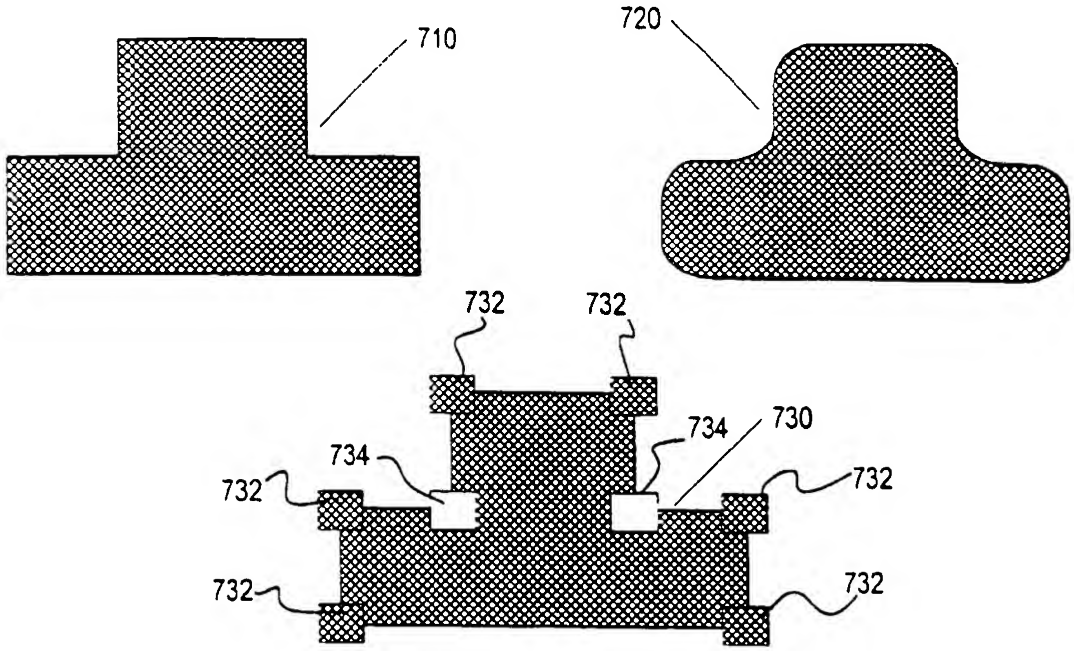


Figure 7(a) (Prior Art)

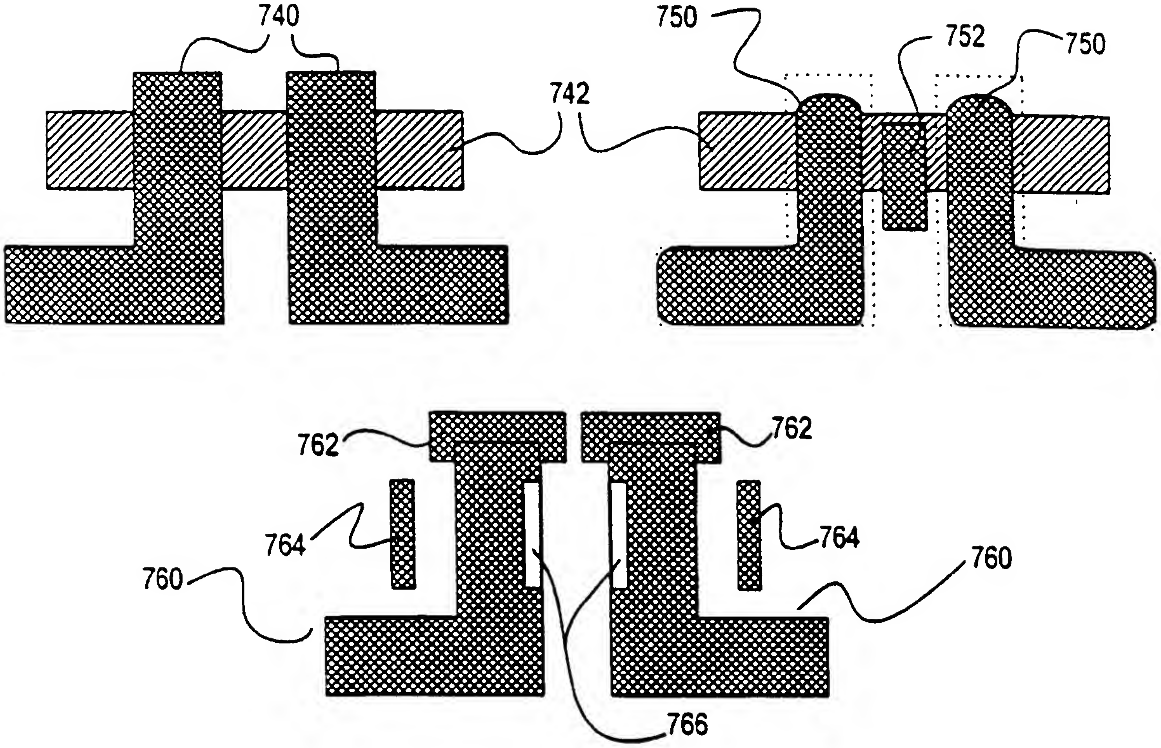


Figure 7(b) (Prior Art)

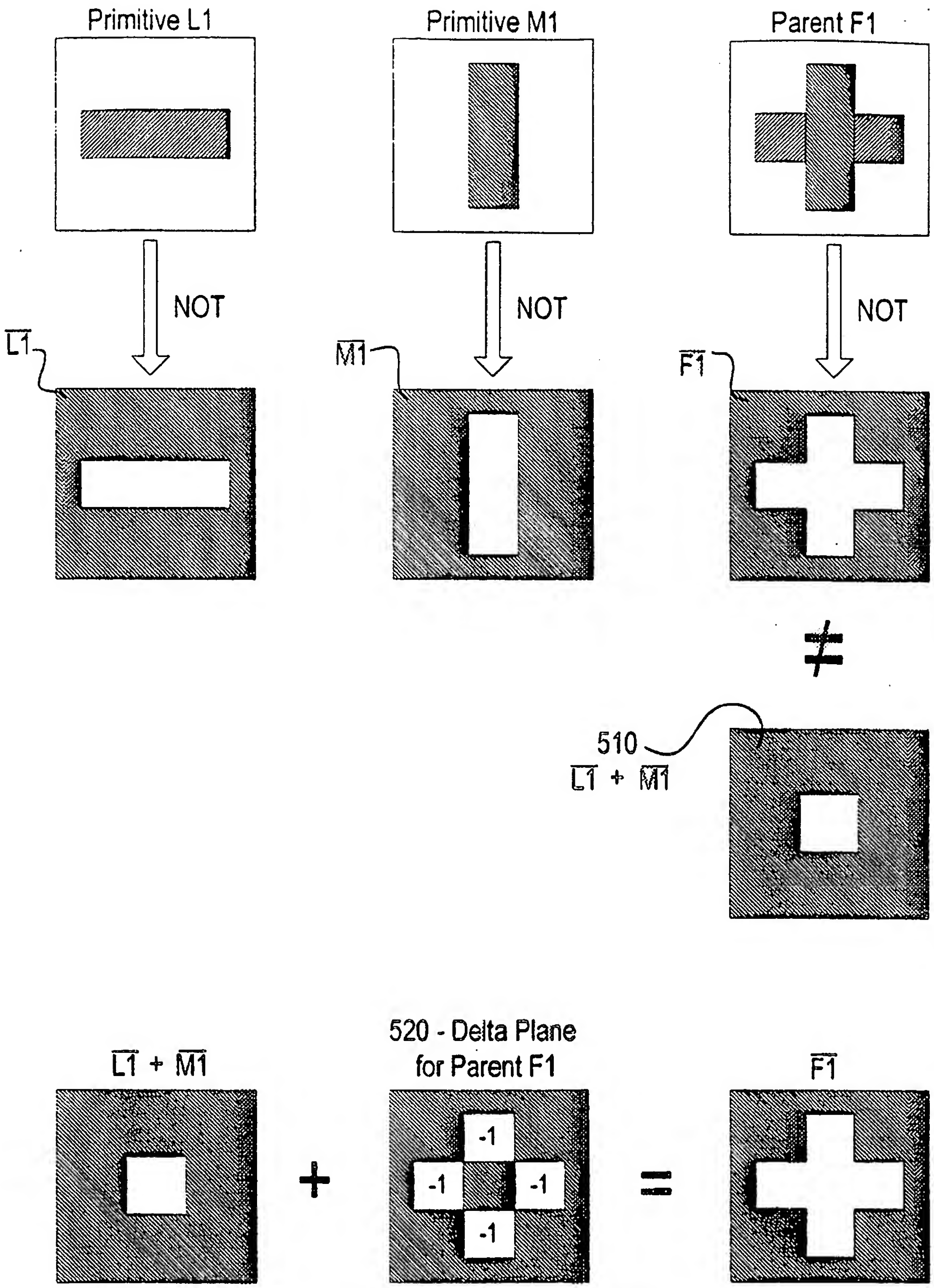

Primitive L1
Primitive M1
Parent F1
NOT
NOT
NOT
$\overline{L1}$
$\overline{M1}$
$\overline{F1}$
≠
510
$\overline{L1}$ + $\overline{M1}$
$\overline{L1}$ + $\overline{M1}$
520 - Delta Plane for Parent F1
-1
-1
-1
-1
+
=
$\overline{F1}$

Figure 5

Generating a Delta Plane for Logical NOT Operation

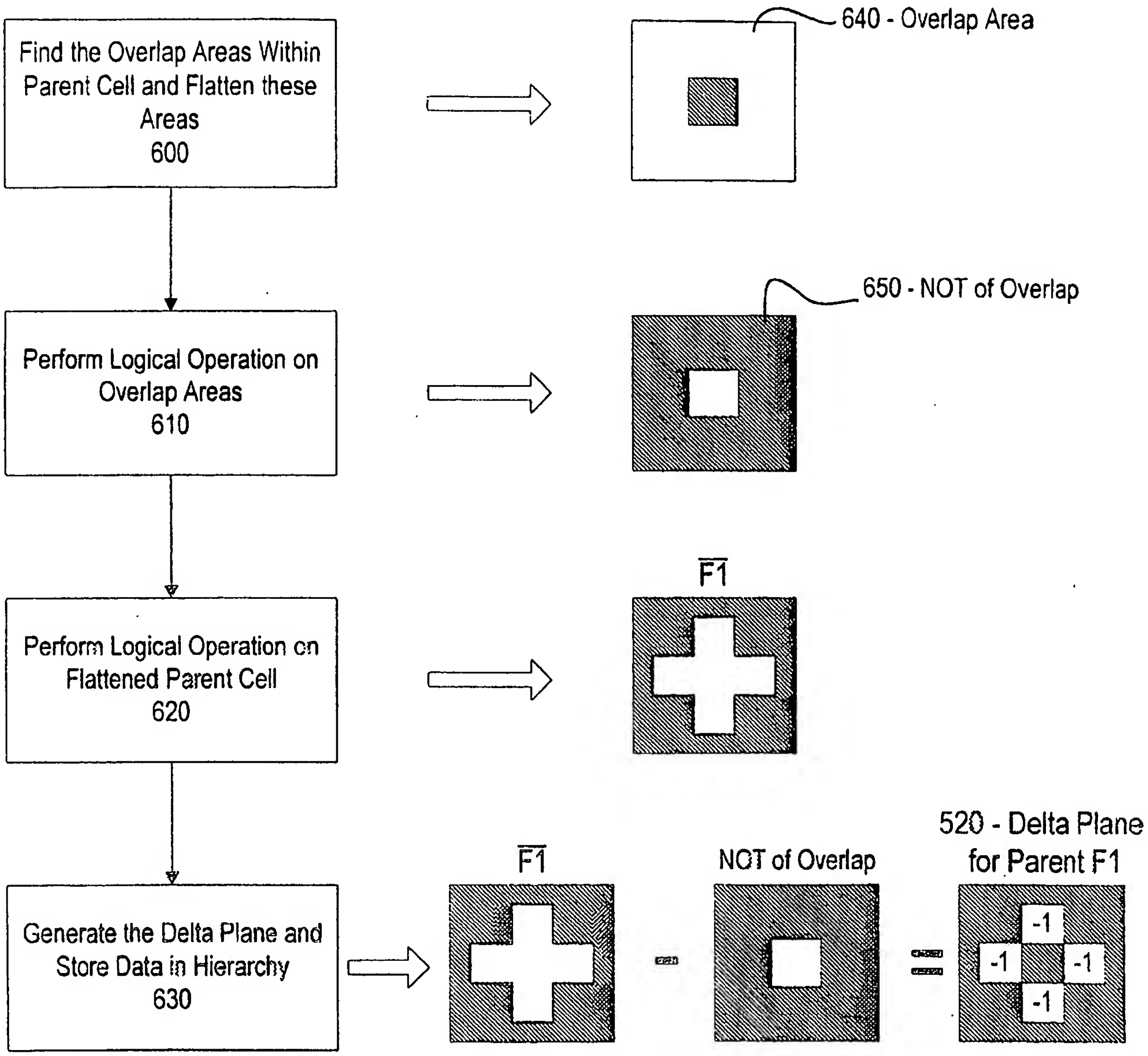

Find the Overlap Areas Within Parent Cell and Flatten these Areas
600
640 - Overlap Area
Perform Logical Operation on Overlap Areas
610
650 - NOT of Overlap
Perform Logical Operation on Flattened Parent Cell
620
$\overline{F1}$
Generate the Delta Plane and Store Data in Hierarchy
630
$\overline{F1}$
-
NOT of Overlap
=
520 - Delta Plane for Parent F1
-1
-1
-1
-1

Figure 6

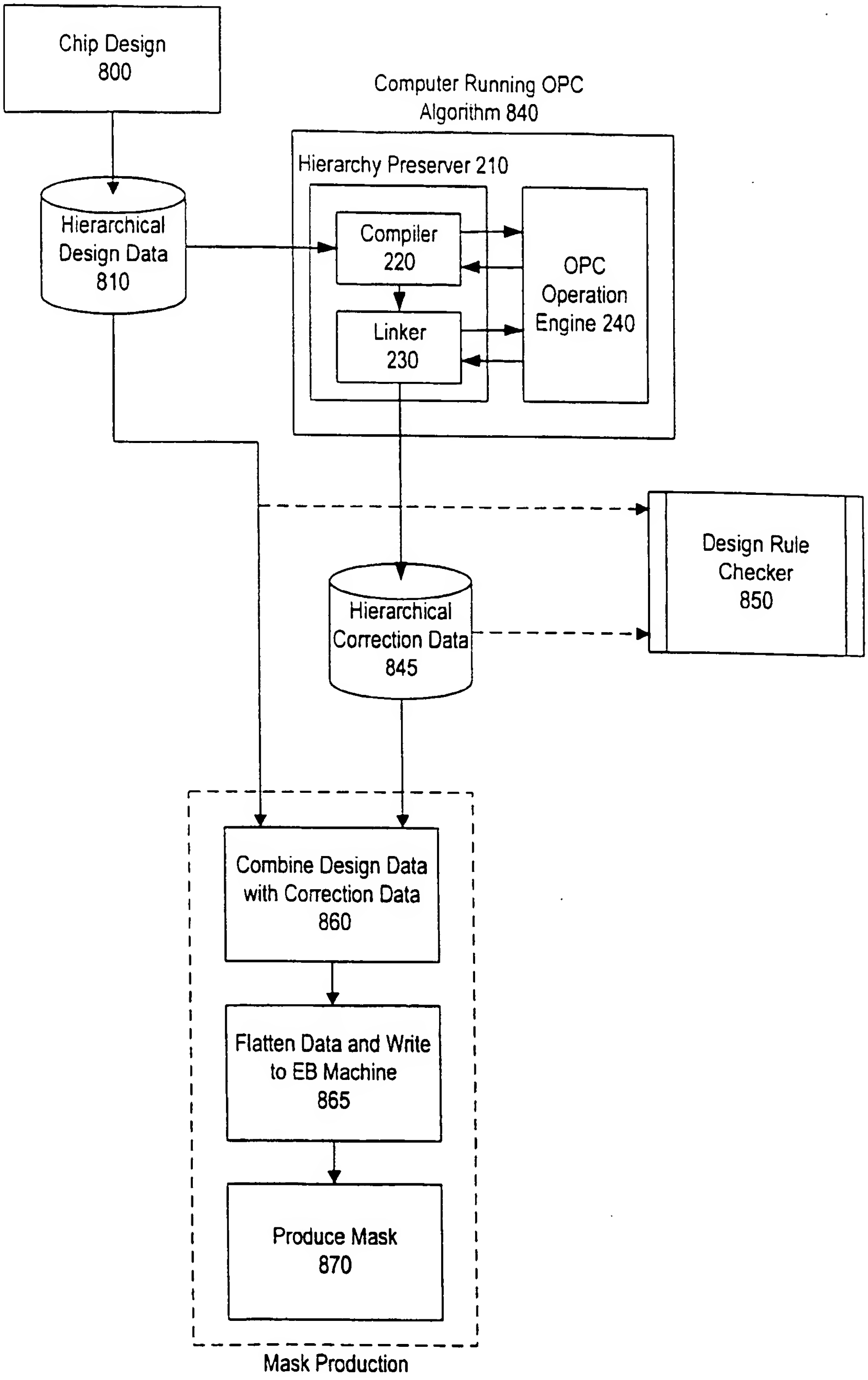

Chip Design
800
Computer Running OPC
Algorithm 840
Hierarchy Preserver 210
Hierarchical
Design Data
810
Compiler
220
Linker
230
OPC
Operation
Engine 240
Design Rule
Checker
850
Hierarchical
Correction Data
845
Combine Design Data
with Correction Data
860
Flatten Data and Write
to EB Machine
865
Produce Mask
870
Mask Production

Figure 8

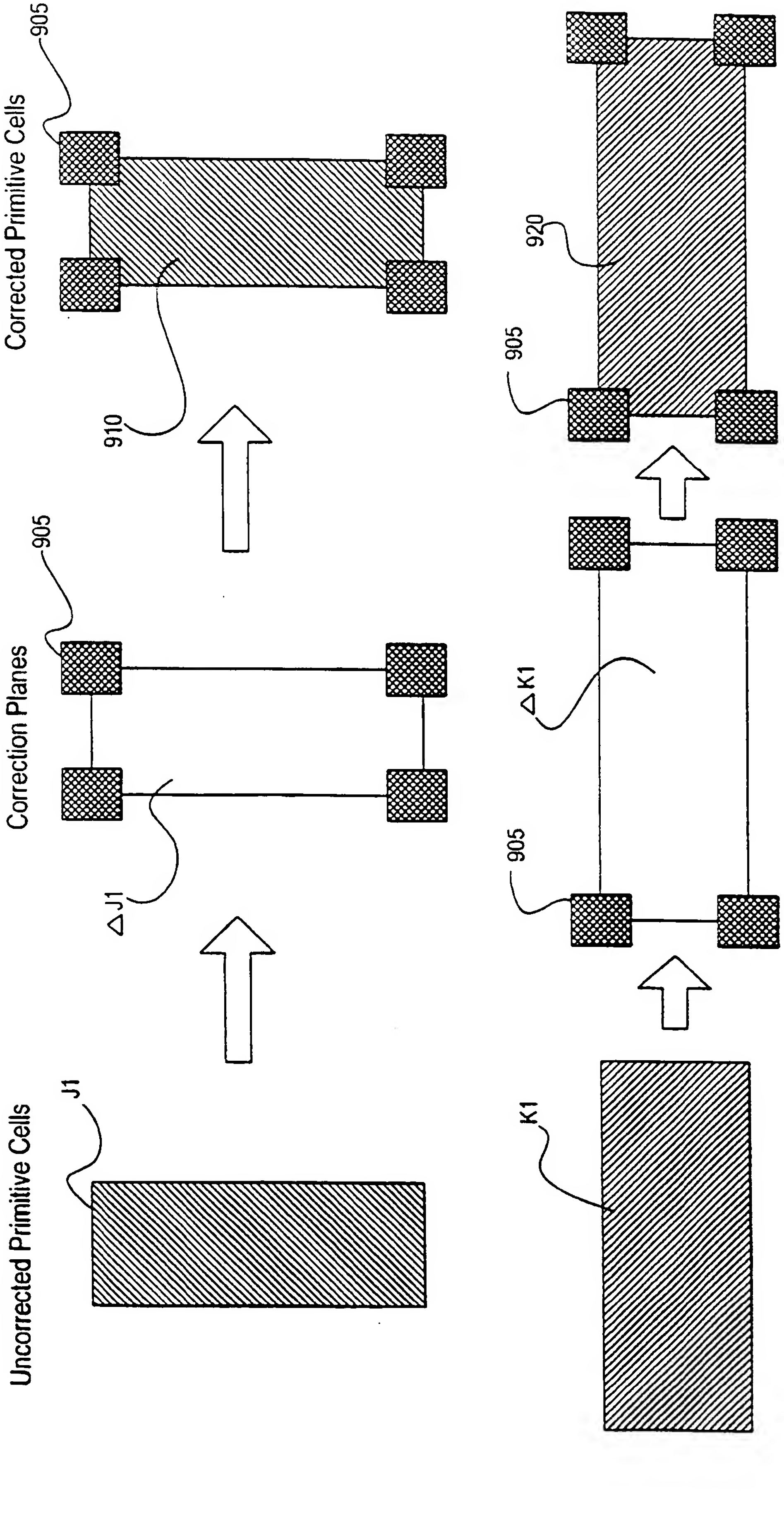

Uncorrected Primitive Cells
J1
K1
Correction Planes
ΔJ1
ΔK1
905
905
905
905
Corrected Primitive Cells
910
920

Figure 9

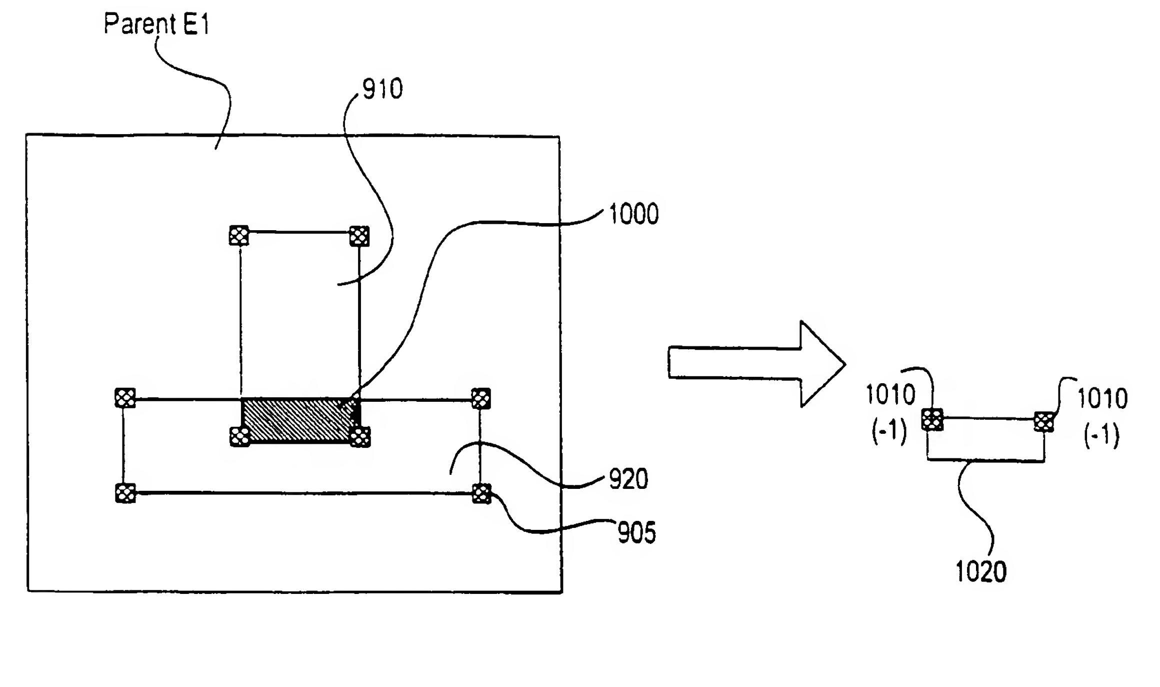


Figure 10(a)

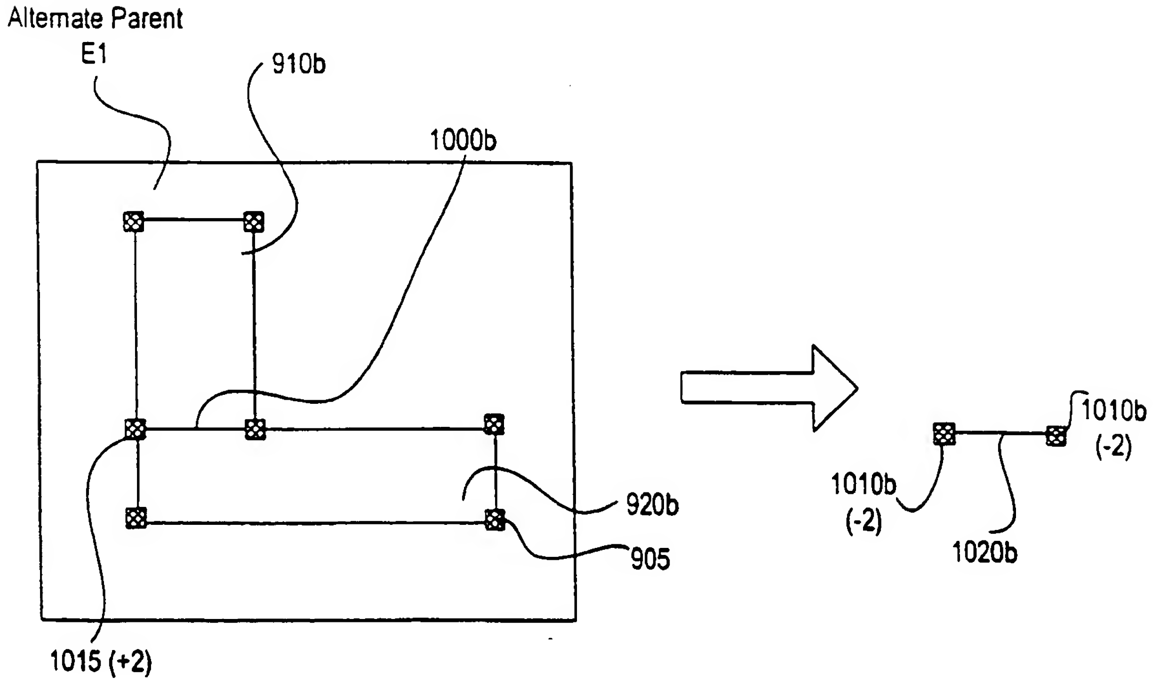


Figure 10(b)

Generating a Delta Plane for OPC

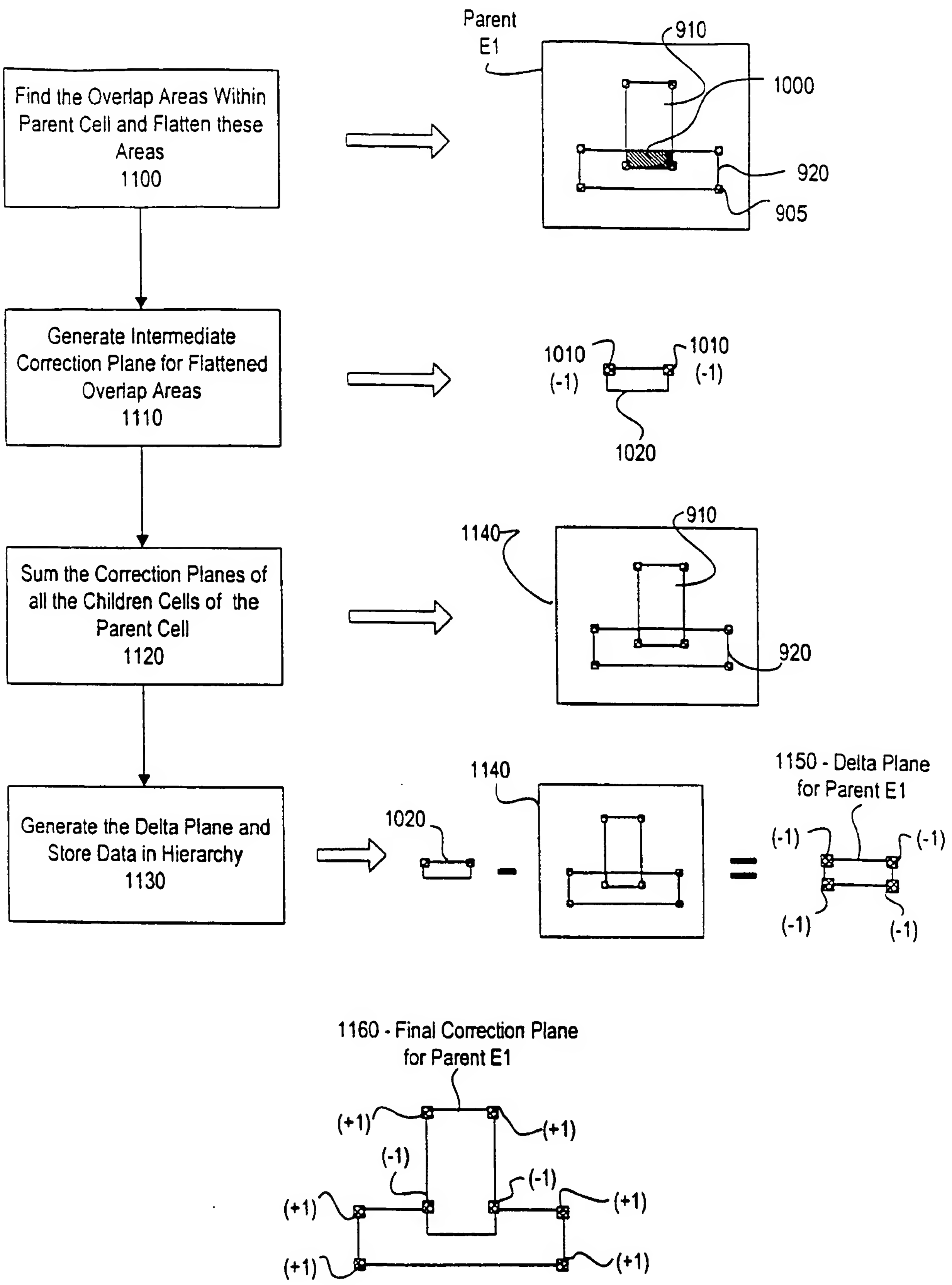

Find the Overlap Areas Within Parent Cell and Flatten these Areas
1100
Parent
E1
910
1000
920
905
Generate Intermediate Correction Plane for Flattened Overlap Areas
1110
1010
(-1)
1010
(-1)
1020
Sum the Correction Planes of all the Children Cells of the Parent Cell
1120
1140
910
920
Generate the Delta Plane and Store Data in Hierarchy
1130
1020
1140
1150 - Delta Plane for Parent E1
(-1)
(-1)
(-1)
(-1)
1160 - Final Correction Plane for Parent E1
(+1)
(+1)
(-1)
(-1)
(+1)
(+1)
(+1)
(+1)

Figure 11

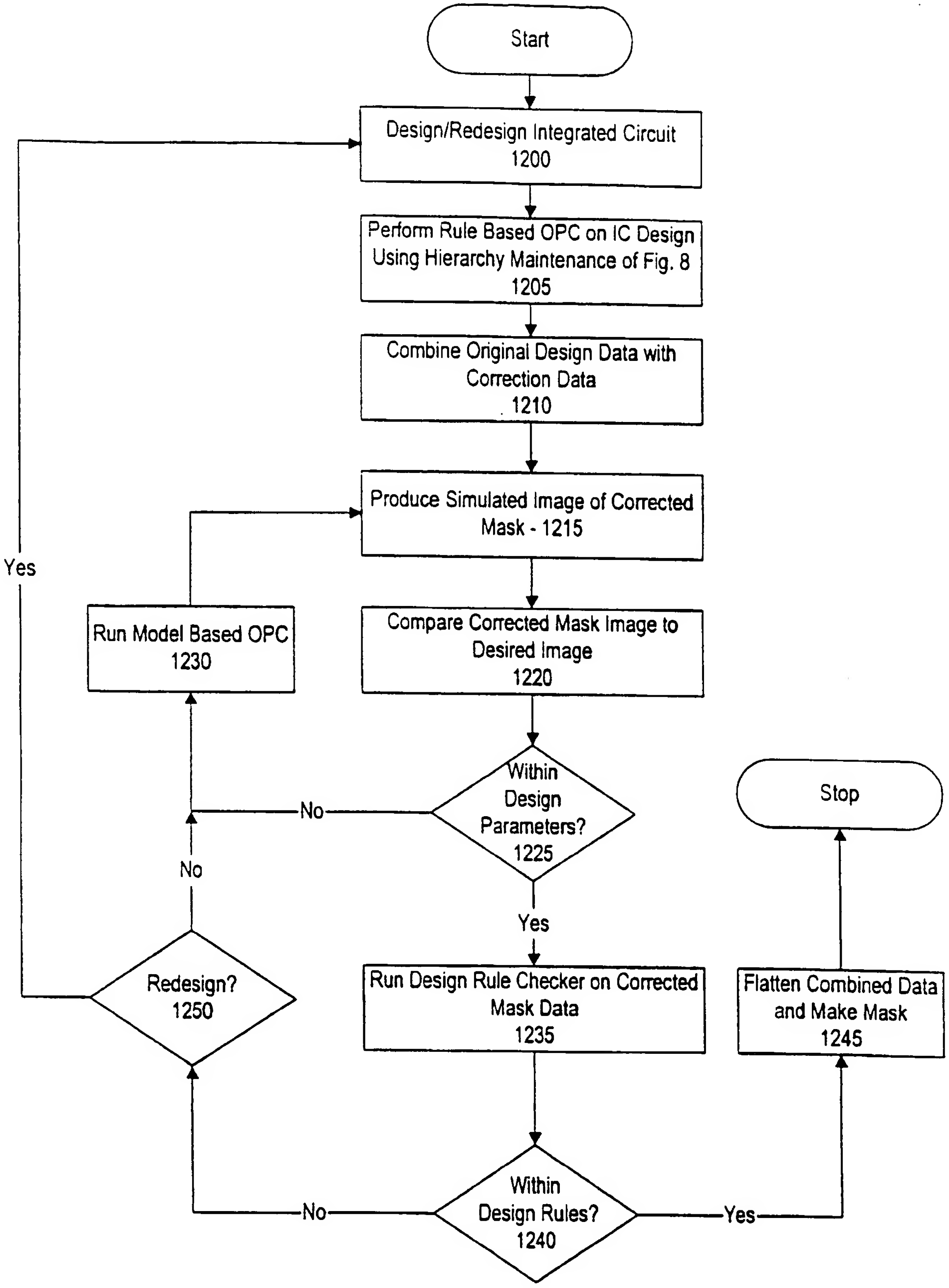
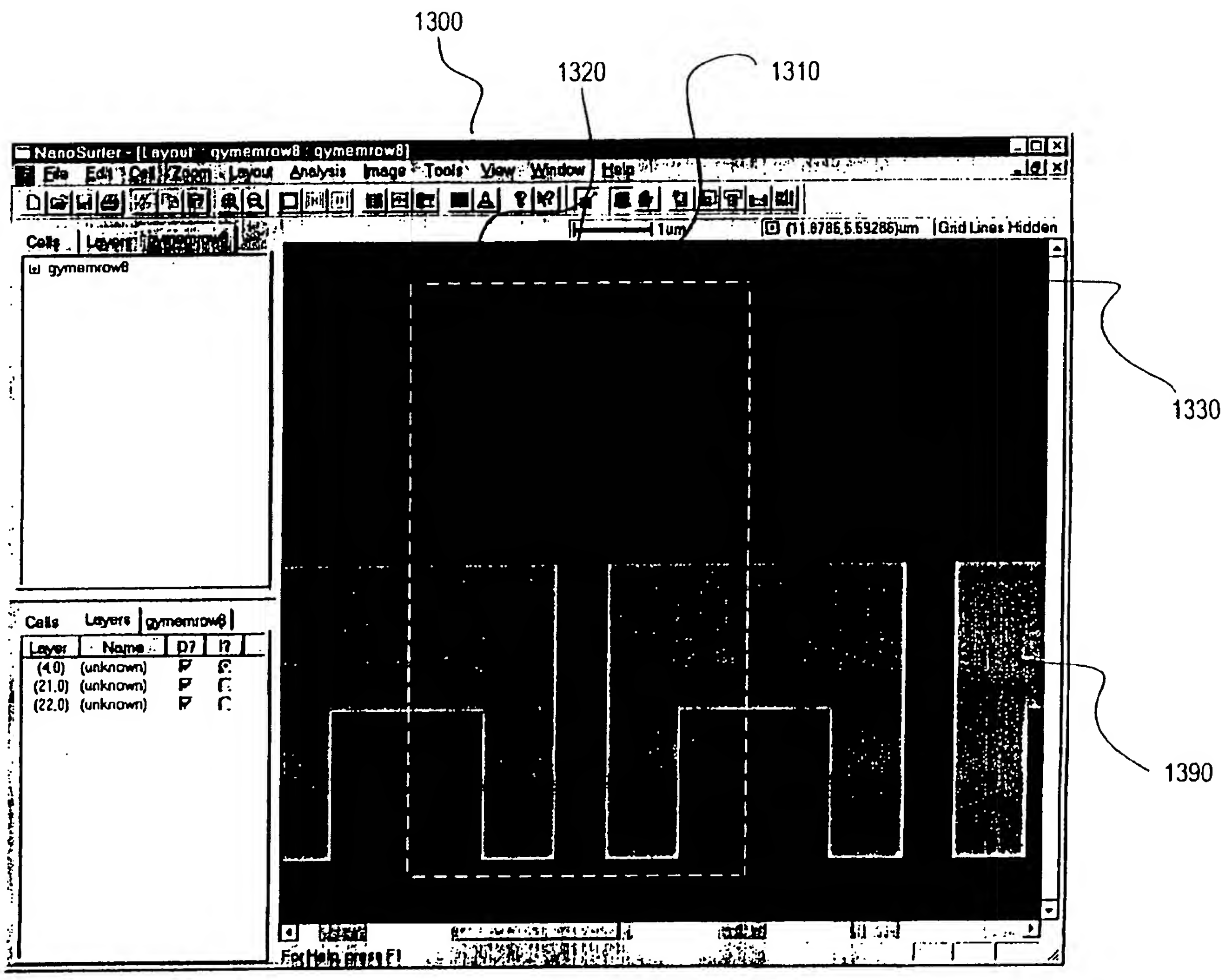

Start
Design/Redesign Integrated Circuit 1200
Perform Rule Based OPC on IC Design Using Hierarchy Maintenance of Fig. 8 1205
Combine Original Design Data with Correction Data 1210
Produce Simulated Image of Corrected Mask - 1215
Compare Corrected Mask Image to Desired Image 1220
Run Model Based OPC 1230
Within Design Parameters? 1225
No
Yes
Stop
No
Yes
Redesign? 1250
Run Design Rule Checker on Corrected Mask Data 1235
Flatten Combined Data and Make Mask 1245
No
Within Design Rules? 1240
Yes

Figure 12



1300
1320
1310
1330
1390
NanoSurfer - [Layout : gymemrow8 : gymemrow8]
File Edit Cell Zoom Layout Analysis Image Tools View Window Help
1um
(11.8786,5.59286)um
Grid Lines Hidden
Cells Layers gymemrow8
gymemrow8
For Help press F1

Cells Layers gymemrow8

Layer	Name	D?	I?
(4.0)	(unknown)	☑	☑
(21.0)	(unknown)	☑	□
(22.0)	(unknown)	☑	□

Figure 13

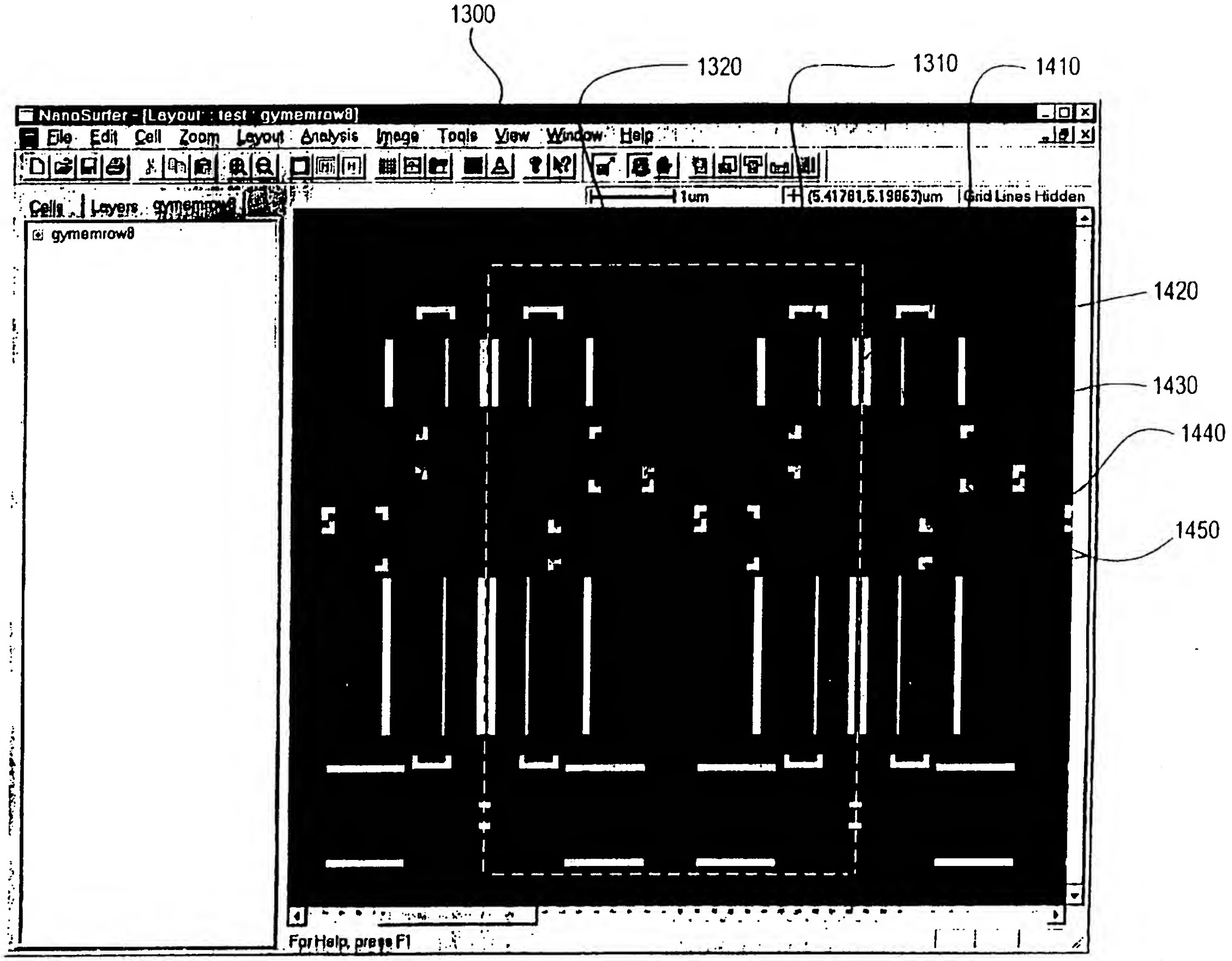
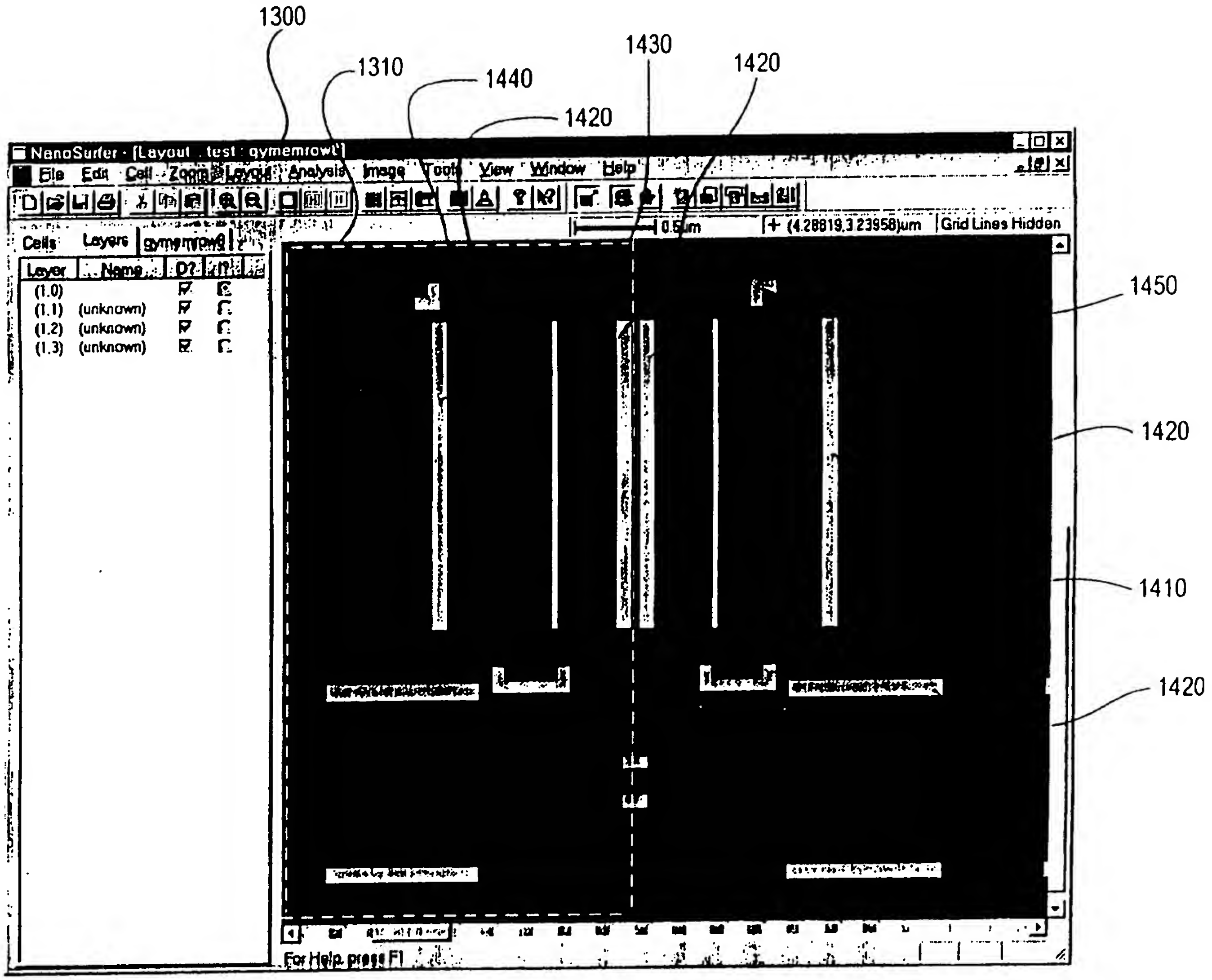

1300
1320
1310
1410
NanoSurfer - [Layout : test : gymemrow8]
File Edit Cell Zoom Layout Analysis Image Tools View Window Help
Cells Layers gymemrow8
gymemrow8
1um
(5.41781,5.19863)um
Grid Lines Hidden
1420
1430
1440
1450
For Help, press F1

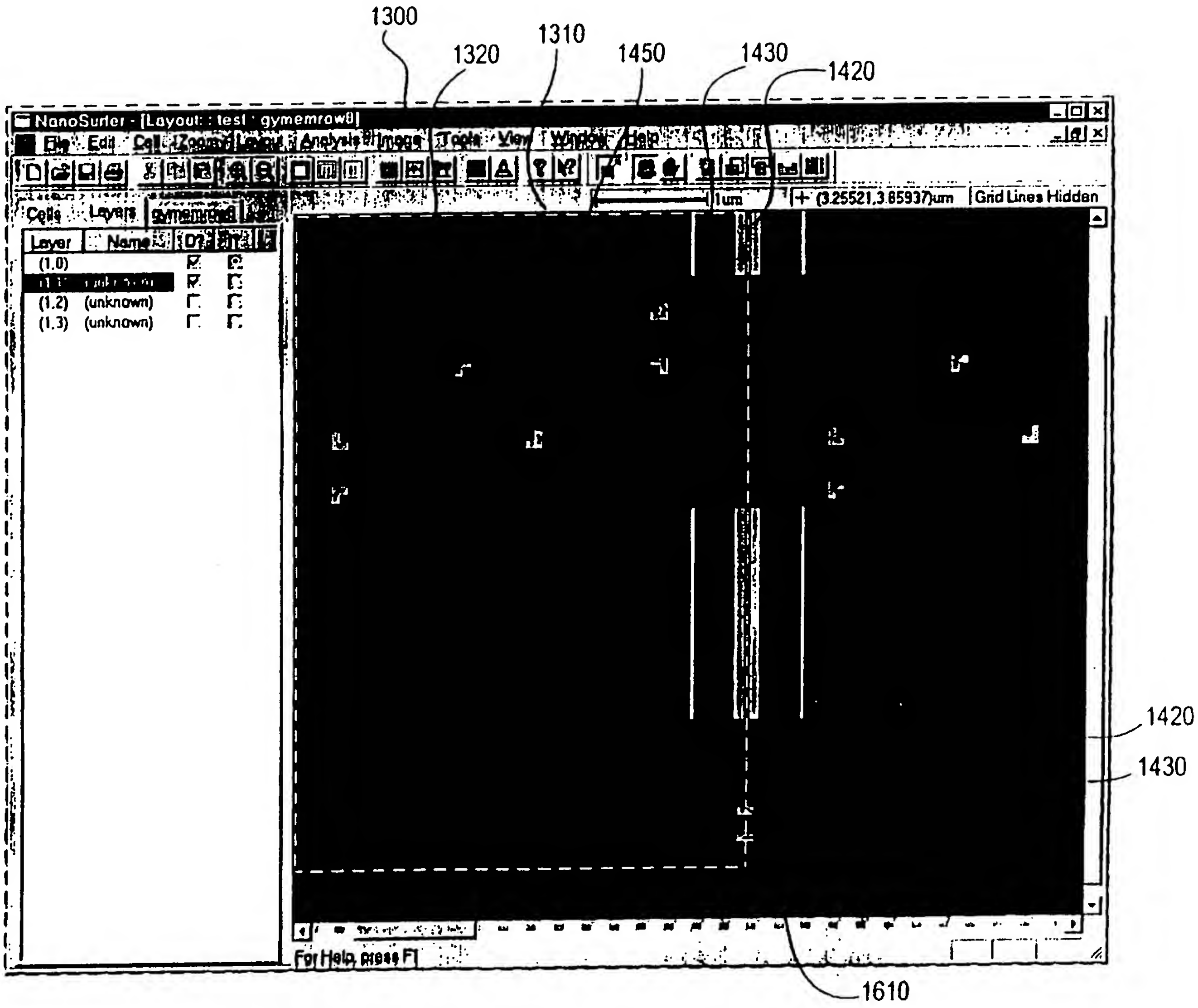
Figure 14


1300
1310
1440
1420
1430
1420
1450
1420
1410
1420
NanoSurfer - [Layout : test : gymemrowt]
File Edit Cell Zoom Layout Analysis Image Tools View Window Help
Cells Layers gymemrow8
0.5µm
Grid Lines Hidden
For Help press F1

+ (4.28819,3.23958)µm

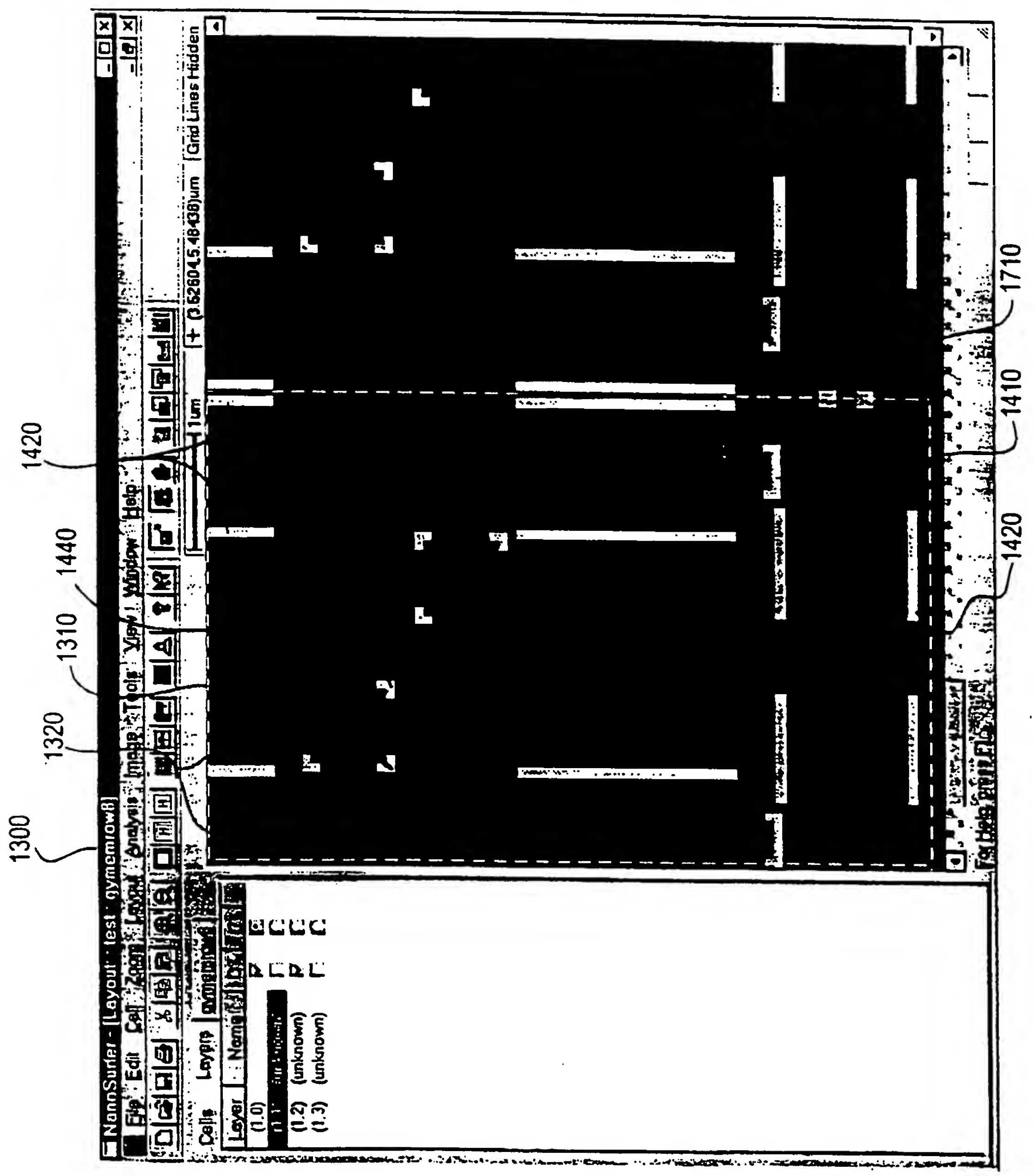
Layer	Name	D?	I?
(1.0)		☑	☑
(1.1)	(unknown)	☑	☐
(1.2)	(unknown)	☑	☐
(1.3)	(unknown)	☒	☐

Figure 15



Layer	Name	D?	?
(1.0)		☑	☑
(1.1)	[illegible]	☑	□
(1.2)	(unknown)	□	□
(1.3)	(unknown)	□	□

Figure 16


NanoSurfer - [Layout : test : gymemrow8]

File Edit Cell Zoom Layout Analysis Image Tools View Window Help

1um

(3.62604,5.48436)um

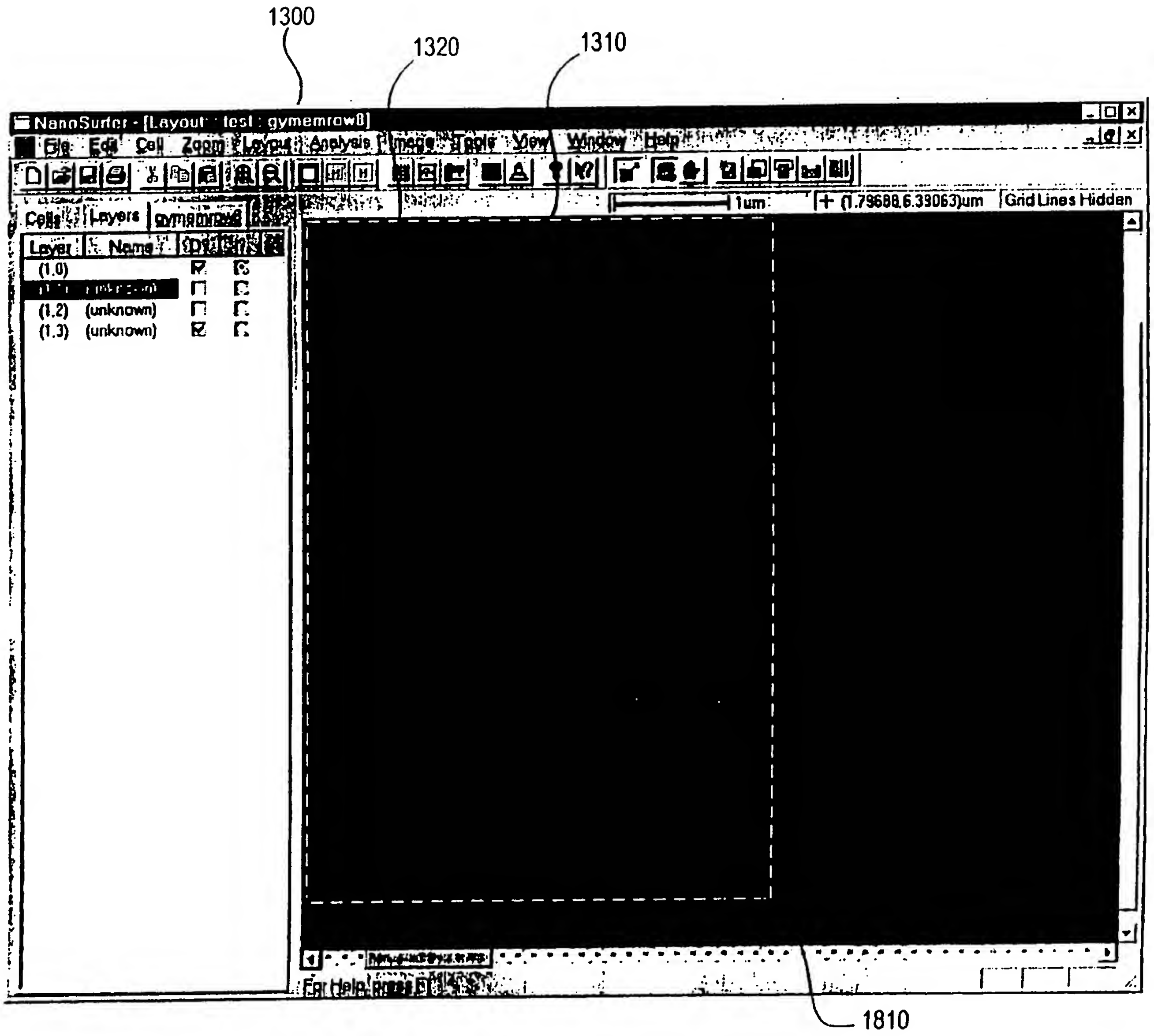
Grid Lines Hidden

Cells Layers

Layer	Name
(1.0)	
(1.1)	(unknown)
(1.2)	(unknown)
(1.3)	(unknown)

For Help, press F1

Figure 17



NanoSurfer - [Layout : test : gymemrow8]

1um

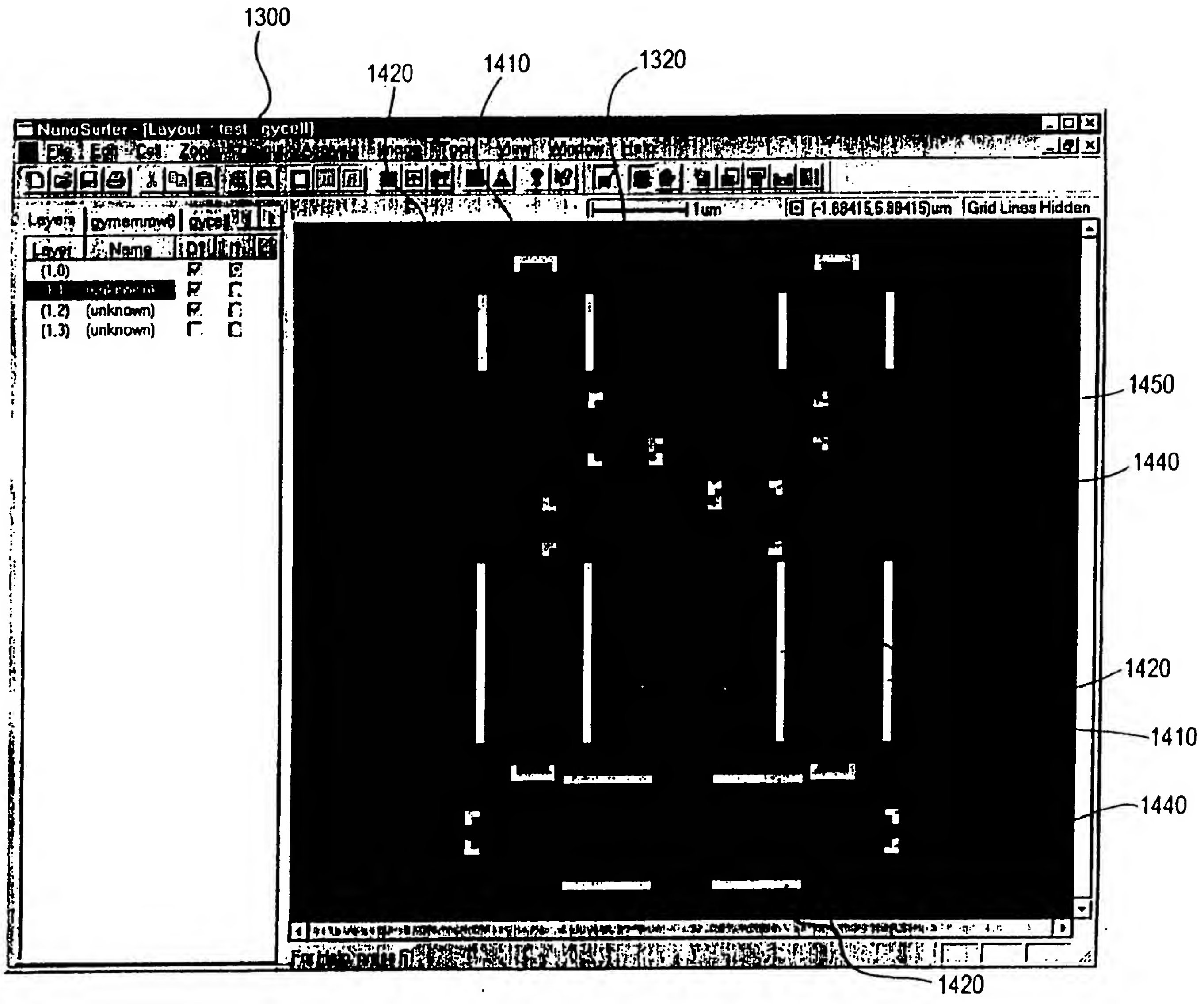
+ (1.79688,6.39063)um

GridLines Hidden

Layer	Name	[illegible]	[illegible]
(1.0)		☑	☐
[illegible]	[illegible]	☐	☐
(1.2)	(unknown)	☐	☐
(1.3)	(unknown)	☑	☐

For Help, press F1

Figure 18



Layer	Name	D	[illegible]
(1.0)		☑	☒
(1.1)	[illegible]	☑	□
(1.2)	(unknown)	☑	□
(1.3)	(unknown)	□	□

Figure 19

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/19439

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G03F 9/00; G06F 17/00, 17/50
US CL : 430/5, 22, 311; 364/488, 489, 490, 491; 395/500
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 430/5, 22, 311; 364/488, 489, 490, 491; 395/500

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y, P	US 5,705,301 A (GARZA et al) 06 January 1998, abstract, summary of the invention, figures 1-2, 3-9, col 4, lines 20-67, col 5, lines 1-36, col 6, lines 33-52, col 8, lines 13-62, col 9, lines 38-65, col 10, lines 14-43, claims 1-7.	1-94
Y	US 5,538,815 A (OI et al) 23 July 1996, abstract, figure 21.	1-94
A, P	US 5,801,954 A (LE et al) 01 September 1998, abstract, summary of the invention, figures 1-8, col 3-6.	1-94
A, P	US 5,702,848 A (SPENCE) 30 December 1997.	1-94
A, P	US 5,740,068 A (LIEBMANN et al) 14 April 1998.	1-94

□ Further documents are listed in the continuation of Box C. □ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search 16 NOVEMBER 1998	Date of mailing of the international search report 01 FEB 1999
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer KEVIN TESKA Joni Hill Telephone No. (703) 305-9704

INTERNATIONAL SEARCH REPORT

International application No. PCT/US98/19439

B. FIELDS SEARCHED

Electronic data bases consulted (Name of data base and where practicable terms used):

APS, IEEE, GPIC

Search terms: optical proximity correction, opc, correct? layout. correct?.data, design rule check###, drc, hierarchic? structure#, hierarchic? tree, layout, gds###